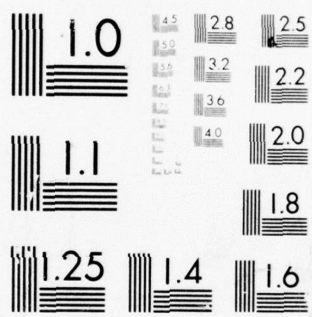
AD-A051 186 ROME AIR DEVELOPMENT CENTER GRIFFISS AFB N Y F/G 17/7
USERS GUIDE FOR ESD LORAN GRID PREDICTION PROGRAM. (U)
DEC 77 S HOROWITZ
UNCLASSIFIED RADC-TR-77-407 NL

1 OF 1
AD
A051186

END
DATE
FILMED
4-78
DDC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(2)

RADC-TR-77-407
IN-HOUSE REPORT
DECEMBER 1977

Users Guide for ESD LORAN Grid Prediction Program

SAMUEL HOROWITZ

Approved for public release; distribution unlimited.

ROME AIR DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
GRIFFISS AIR FORCE BASE, NEW YORK 13441

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

APPROVED:

Edward A. Lewis

EDWARD A. LEWIS, Chief
Propagation Branch
Electromagnetic Sciences Division

APPROVED:

Allan C. Schell

ALLAN C. SCHELL, Acting Chief
Electromagnetic Sciences Division

FOR THE COMMANDER: John P. Huss

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

Field	Entry
1. REPORT NUMBER	(14) RADC-TR-77-407
2. GOVT ACCESSION NO.	
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle)	(6) USERS GUIDE FOR ESD LORAN GRID PREDICTION PROGRAM
5. TYPE OF REPORT & PERIOD COVERED	In-house
6. PERFORMING ORG. REPORT NUMBER	
7. AUTHOR(s)	(10) Samuel Horowitz
8. CONTRACT OR GRANT NUMBER(s)	
9. PERFORMING ORGANIZATION NAME AND ADDRESS	Deputy for Electronic Technology (RADC/EEP) Hanscom AFB Massachusetts 01731
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS	61102F (16) 2305J201 (17) J2
11. CONTROLLING OFFICE NAME AND ADDRESS	Deputy for Electronic Technology (RADC/EEP) Hanscom AFB Massachusetts 01731
12. REPORT DATE	(11) De[redacted]77
13. NUMBER OF PAGES	86
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	
15. SECURITY CLASS. (of this report)	Unclassified (12) 87p.
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	
16. DISTRIBUTION STATEMENT (of this Report)	Approved for public release; distribution unlimited.
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)	
18. SUPPLEMENTARY NOTES	
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)	L. F. propagation Propagation over irregular ground L. F. navigation LORAN C/D

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The purpose of the LORAN Grid Prediction Computer program is to transform geographic coordinates into a LORAN time difference (TD). Utilizing available map data of terrain and lithology and applying an integral form of Maxwells' equations, the propagation time of a low frequency pulse over irregular and inhomogeneous ground can be calculated. The method of data tape production and TD calculation is described with program listings. The technique has been successfully used by several U. S. Government agencies.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

1

309 050

Alu

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

BLANK PAGE

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

Preface

This work was performed at the request of the LORAN Systems Project Office, Electronic Systems Division, Hanscom AFB, MA 01731.

The type of program described in this paper involved numerous people for its success, in particular: Dr. Terence J. Elkins, RADC/EEP, Capt. Randolf Gressing, ESD/DCL, and Mr. William McComish, Boston College. Deeply appreciated are their vigorous and indispensable technical support.

PRECEDING PAGE BLANK - NOT FILMED

Contents

1. INTRODUCTION 7

2. SYSTEM CONFIGURATION 8

3. DESCRIPTION OF PART III 9

3.1 Overview 9
3.2 Prediction Program Flow Chart 10
3.3 Data Input and Output Setup 13
3.3.1 Input 13
3.3.2 Output 15
3.4 Program Subroutines and Functions 15
3.5 Subroutine Description 16
3.5.1 LORANCO 16
3.5.2 Subroutine CORRAD 17
3.5.3 Subroutine GEOPTS 18
3.5.4 Subroutine GEORET 20
3.5.5 Subroutine GETELV 21
3.5.6 Subroutine SETUP 21
3.5.7 Subroutine INT 21
3.5.8 Subroutine UNPACK 22
3.5.9 Subroutine INEQ 22
3.6 Program Listing for Part III 24
3.7 Sample Test Run 46
3.8 Output Listing 47

4. DESCRIPTION OF PART II 48

4.1 Overview 48
4.2 Subroutine Description 50
4.2.1 Program CONIMP 50
4.2.2 Subroutine SETUP 50
4.2.3 Subroutine REPK 51
4.2.4 Subroutine GRIM 51

Contents

4.3 Resistivity Tables	51
4.4 Program Listing for Part II	56
5. DESCRIPTION OF PART I	60
5.1 Overview	60
5.2 Program Listing for Part I	62
6. DESCRIPTION OF PART IV	71
6.1 Overview	71
6.2 Program Listing for Part IV	72
REFERENCES	83
CONDUCTIVITY BIBLIOGRAPHY	85

Illustrations

1. Total LORAN Grid Prediction System	8
2. LORAN Prediction Package – Part III	9
3. LORAN Program Flow Chart	11
4. ESD LORAN Grid Prediction Program	12
5. Deck Setup for Program LORANCO	14
6. Geodesic Geometry	18
7. Data Base of Ground Electrical Properties – Part II	49
8. Ground Model for Surface Impedance Calculators	49
9. Formation of Data Base – Part I	61

Tables

1. Ground Resistivity Values for Typical Soils	51
2. Rock Types and Resistivities	52

Users Guide for ESD LORAN Grid Prediction Program

1. INTRODUCTION

The ability to precisely deliver ordnance or men and material under all weather conditions in an adverse battlefield environment, is one of the most severe requirements imposed on U. S. Worldwide Tactical Air Forces. A key element in meeting this requirement involves obtaining and providing position data in a specified coordinate system, to allow navigation of tactical aircraft to desired locations with sufficient accuracy for rendezvous or weapon release. The LORAN radio navigation system is being relied upon more and more as a principle source of navigation information in tactical airborne operations. Position indication is given in terms of LORAN TD's (time difference) which, because of propagation anomalies, do not correspond precisely to earth fixed geodetic coordinates. Therefore, each LORAN chain requires a grid prediction for its coverage area. Such a grid prediction computer program package has been developed at RADC/EEP and is described herewith. This manual contains sufficient information to enable the experienced programmer to understand the programming aspects of LORAN grid prediction and includes a detailed functional description and its operation.

(Received for publication 8 December 1977)

2. SYSTEM CONFIGURATION

The operation of the entire LORAN Grid Prediction System from obtaining the required input data off maps to the calculation of the output time difference is illustrated in Figure 1. The system has been divided into the four following parts:

PART I: Formation of map digitized.

PART II: Calculation of surface impedance data base.

PART III: Field LORAN prediction package.

PART IV: Updating data base by comparison of calculated and experimental results.

A sufficient description of the various computer program is presented so that the user can obtain the required LORAN coordinates. The data base is generated in PARTS I and II and updated when measured data is available in PART IV. In PART III of the systems, the main set of calculations are performed and this is described first in Section 3. The technique for translating the earth's electrical properties into a surface impedance and properly sequenced onto a rapid access magnetic disc is described in Section 4. Brief description of map data digitization and system tuning or data base updating is given in Sections 5 and 6, respectively.

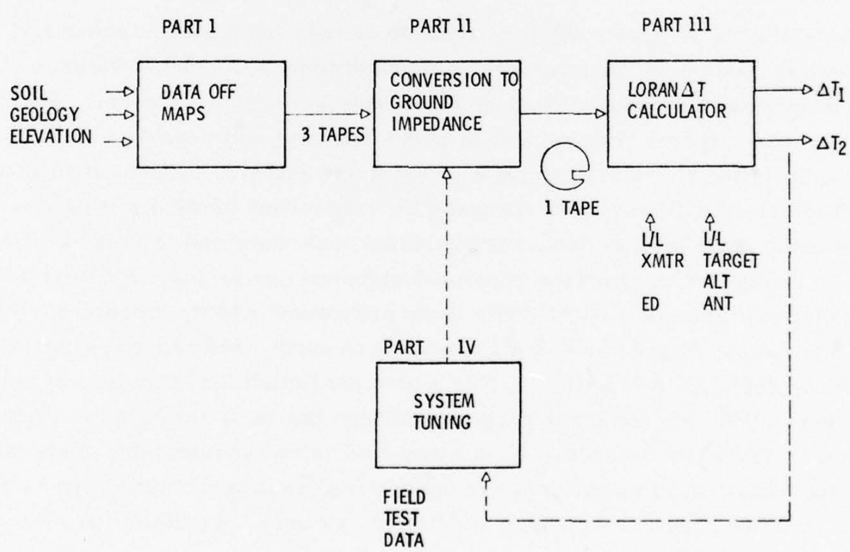


Figure 1. Total LORAN Grid Prediction System

3. DESCRIPTION OF PART III

3.1 Overview

A block diagram of the PART III LORAN prediction package is shown in Figure 2. Its purpose is to furnish the LORAN TD's for a desired target given the ground properties of the system coverage area. For each LORAN chain, the geodetic location of the master and two slave transmitters must be known in addition to the two slave emission delays. A magnetic tape of the ground electrical properties for the given coverage area is also required. The latitude and longitude of the desired target and delivery altitude are inserted into the program and a time of arrival (TOA) from each of the transmitters is computed. Subtracting the master TOA from each of the slave TOA's yields two TD's which determine the LORAN coordinates of the target.

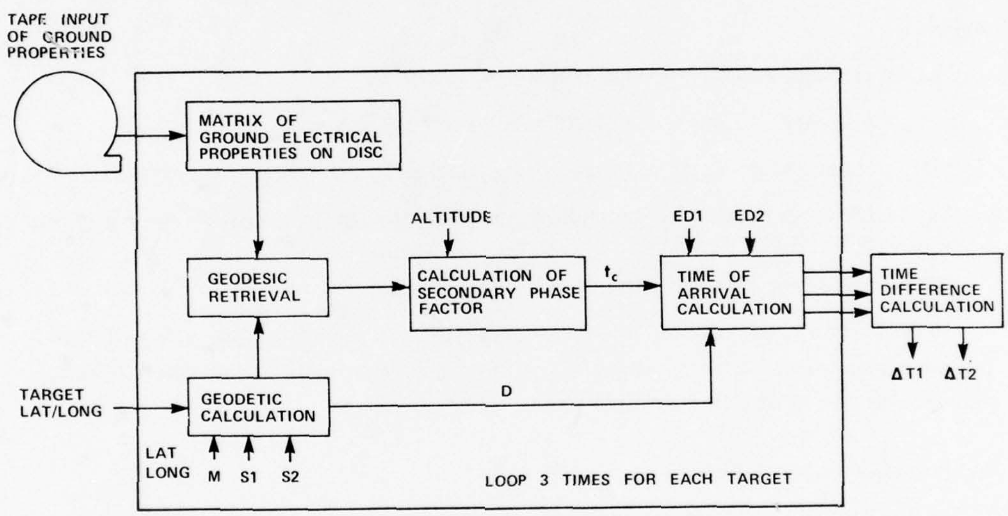


Figure 2. LORAN Prediction Package – Part III

The operation of this program can be followed by referring to Figure 2. The input tape contains the ground electrical properties which consist of elevation and complex surface impedance for the area of interest. This is in the form of a matrix of data points every 30 arc sec in latitude and longitude. The information is recorded onto a disc for rapid access of data between any two points in the service area as required by the geodesic retrieval.

The input data cards contain the following: (a) transmitter coordinates with associated emission delays, and (b) target coordinates, delivery altitude, integration step size, and type of receiving antenna for each target. This data is first used in the geodetic calculation to determine path length (D) between each of the transmitters and the target, and to determine points along the geodesic path governed by the distance increment or integration step size. The points along the geodetic path are input to the geodesic retrieval, which in turn obtains the ground electrical properties of the points from the matrix on the disc. This information is used to calculate the time correction or secondary phase factor due to the decrease in propagation velocity, compared to free space, when a signal propagates over the earth's surface. Time of arrival calculations can then be made from the following equation:

$$TOA = \frac{n}{c} D + t_c + ED \qquad (1)$$

where

n = atmospheric index of refraction = 1.000338.

c = velocity of light = 2.997925×10^8 meters/second.

D = length of geodetic path from transmitter to receiver in meters.

t_c = time correction or secondary phase factor for propagation over a given path length in μsec.

ED = emission delay in μsec.

Three such calculations, one from each transmitter to target, are required for each prediction. Subtracting the master TOA from each of the slave TOA's, one obtains the LORAN coordinate prediction.

3.2 Prediction Program Flow Chart

A flow chart illustrating the operation of the LORAN Grid Prediction package is illustrated in Figure 3 and subroutine relationships to the driving program LORANCO is shown in Figure 4. This package consists of a deck with approximately 1300 cards and requires a core memory of 120K base eight (8). The three transmitter (M, S1, S2) coordinates with corresponding emission delays are read into the program from the input data deck. For computation purposes, the geographic coordinates are converted into radians by a call to subroutine CORRAD. The target coordinates are then read into the program with corresponding information on delivery altitude in meters, computation step size in kilometers, and type of receiving antenna (Electric dipole = 1, Loop = 0). Similarly, these coordinates are converted to radians. The program is terminated when the step size (ADELS)

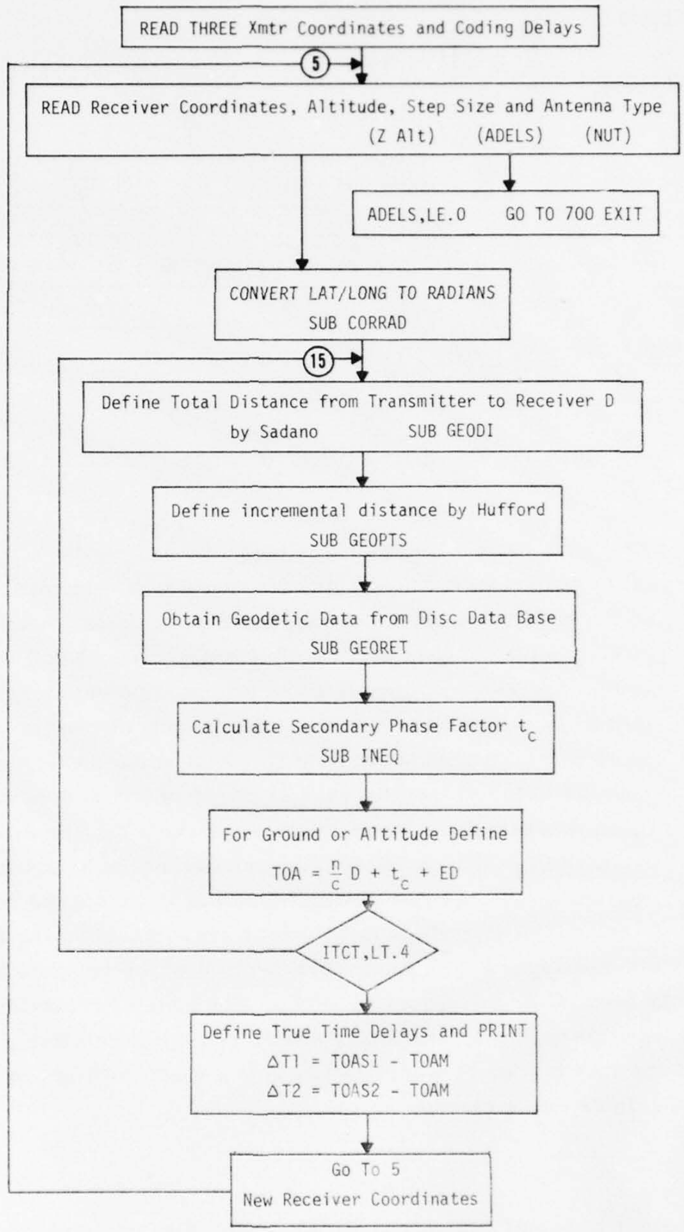


Figure 3. LORAN Program Flow Chart

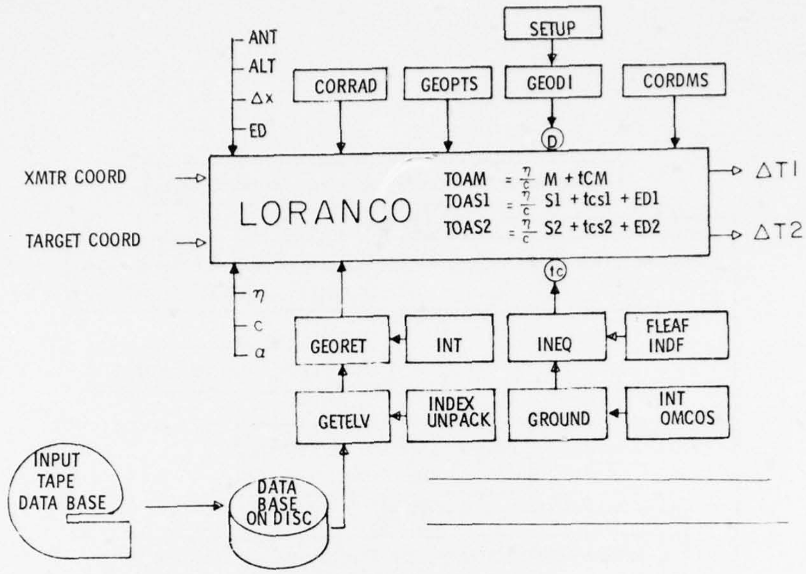


Figure 4. ESD LORAN Grid Prediction Program

is equal to or less than zero. Therefore, a blank card after the last desired target card causes the program to exit. With information on transmitter and target location, the program can now determine the propagation path. The total geodesic distance for primary wave delay is calculated in subroutine GEODI which uses the Sodano formulation.[1, 2] The coordinates of incremental points along the propagation path are defined in subroutine GEOPTS. The ground electrical properties for these points are obtained through subroutine GEORET which in turn calls subroutine GETFLV. In the latter subroutine, the required points are indexed to address data for the random access disc. Upon return from the disc, the data is unpacked and returned to GEORET. With a detailed description of the points along the propagation paths available, one can now calculate the secondary phase factor or time correction, t_c, due to the disturbing influence of the earth using the Hufford Integral formulation equation.[3, 4, 5] Ninety-five percent of the compute time is required for this calculation in subroutine INEQ. The time of arrival is the sum of the primary wave travel time, secondary phase factor, and emission delay. All the required information for this calculation over a given path is now available and the resultant calculation is stored.

Due to the large number of references on this page, the references will not be footnoted. See references, page 83.

The above procedure is repeated three times, once from each transmitter to receiver by looping back to statement 15 after each completed TOA. With the completion of the TOA calculations, the predicted time difference for a given geographical coordinate is calculated (T, T2) in the driving program LORANCO and output printed. Control is then transferred back to statement 5 where new target coordinates are read into the system, and the entire process is repeated until ADELS is made equal to or less than zero.

Figure 4 is an additional flow chart illustrating the subroutine relationships to the driving program LORANCO.

3.3 Data Input and Output Setup

3.3.1 INPUT

Program LORANCO requires a data deck and data tape for operation loaded as shown in Figure 5. The data deck supplies the geographic location of transmitters and targets and additional required constants such as emission delay, altitude, step size, and type of receiver antenna. The data tape supplies the ground electrical properties for the entire service area covered by the transmitters.

The first three cards in the data deck describe the transmitter input data as follows:

Cols	Data	Format
1-2	Blank	
3-6	Alpheric numeric identifier for transmitter	3A8
26-42	Latitude data	I5, I3, F7.3, A1
26-30	Latitude, degrees	I5
31-33	Latitude, minutes	I3
34-41	Latitude, seconds	F7.3
42	Latitude, N or S	A1
43-58	Longitude, data	I5, I3, F7.3, A1
43-47	Longitude, degree	I5
48-50	Longitude, minutes	I3
51-57	Longitude, second	F7.3
58	Longitude, E or W	A1
59-78	Emission Delay	F20.3

All degrees, minutes, and seconds are right justified in their respective field. Emission delay is in units of μsec. The order of transmitter cards are master, slave 1, and slave 2. The master emission delay will always be zero. Slave 1 emission delay is always less than that of slave 2.

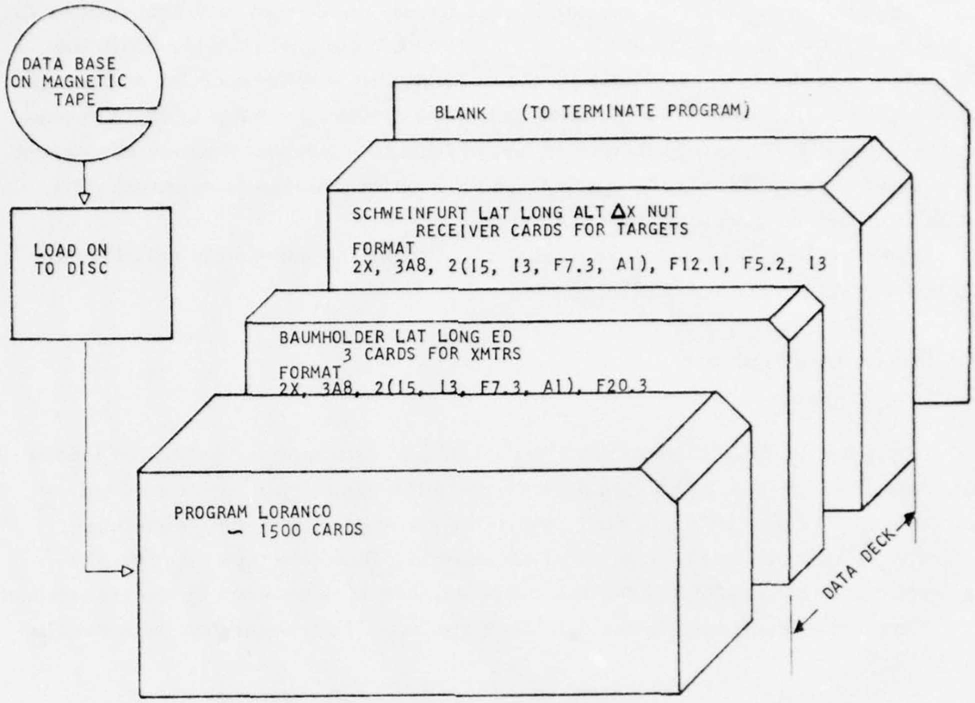


Figure 5. Deck Setup for Program LORANCO

The next set of cards contain the target input data. It is read in as one target per card so there will be as many receiver cards as targets with the following format:

Cols	Data	Format
1-2	Blank	
3-26	Alpheric numeric identification for receiver identification	3A8
26-42	Latitude data	I5, I3, F7.3, A1
26-30	Latitude, degrees	I5
31-33	Latitude, minutes	I3
34-41	Latitude, seconds	F7.3
42	Latitude, N or S	A1
43-58	Longitude data	I5, I3, F7.3, A1
43-47	Longitude, degrees	I5
48-50	Longitude, minutes	I3

Cols	Data	Format
51-57	Longitude, seconds	F7.3
58	Longitude, E or W	A1
59-70	Altitude, meters	F12.1
71-75	Step Size, kM	F5.2
76-78	Type of Antenna 0 or 1	I3

The aircraft altitude at the release point is specified in meters and the distance increment or step size in kilometers typically 0.5 kilometers. NUT defines the type of receiver antenna. For a vertical antenna, a 1 is placed in column 78, for a loop, a 0 in column 78. A blank card or zero in columns 71-75 will terminate the program.

The data tape contains information on the electrical properties of the ground and covers the entire service area. Points selected outside this area will cause the program to print "OUT OF ACCEPTABLE RANGE, FURTHER CALCULATIONS FOR THIS PATH HAVE BEEN DELETED." The supplied magnetic tape contains 5760 records of 60 words each with each word representing 120 data points or one degree of latitude. Each data point is defined by a complex surface impedance and an elevation. The current area covered is 66 to 14 degrees in longitude and 48 to 54 degrees in latitude. This tape is read into the random access data base disc unchanged in format with the sequential record number on the tape becoming the random access index array address number. No operation on the tape is required.

3.3.2 OUTPUT

Program LORANCO produces the following printed output:

(a) For each transmitter, an echo printout of columns 3-58, of the input card.

(b) For each receiver, an echo printout of columns 3-58 of the input card.

(c) For each transmitter to receiver path, geodesic path information, and a printout of the parameters used by subroutine INEQ.

(d) Results of the ground wave time delay calculations in the form of the list NAMI. The list NAMI contains information on time of arrivals (TOA), time delays (TD1, TD2), emission delays (ED1, ED2), geodesic distances (DISTSOD), primary wave times (TPW), and secondary phase factor times (TIMDUM).

3.4 Program Subroutines and Functions

LORANCO - Reads inputs, defines three paths, calls data base and INEQ, calculates TOA and TD.

CORRAD - Degrees (Lat, Long) to radians.

CORDMS - Radians to degrees (Lat, Long).

GEOPTS - Defines points along geodetic from transmitter to receiver.

INT	- Calculate first and second derivative of elevation.
GETELV	- Read geophysical data off disc.
UNPACK	- Unpack data from disc.
GEORET	- Returns geophysical data to driving program.
INEQ	- Solution of integral equation for secondary phase factor.
INDF	- Induction field calculation (E or H field).
CNEUKEN	- Interpolation routine for integration.
GEODI	- Calculation of total geodetic distance by Sodano and back azimuth.
SETUP	- Constants for spheroid to be used.
GROUND	- Introduces variable ground impedance and variable ground terrain into integral equation formulation of the ground wave.
CANG	- Calculates argument of a complex number.
INDEX	- Calculates index values of data base variables from LAT/LONG.
WERF	- Calculates error function.
OMCOS	- Calculates 1 - cos (X).
FLEAF	- Calculates ground wave attenuation function over flat ground using flat earth theory.

3.5 Subroutine Description

3.5.1 LORANCO

This is the driving module for the entire program. It reads the input coordinates of both transmitters and receivers, defines the geodesic path, receives data base parameters and calculates the secondary phase factors, time of arrivals, and time difference. The first three read cards, one for each station, furnish the transmitter locations and associated emission delays. A call to CORRAD changes the units of the input data from degrees to radians. The first target or receiver geographic coordinates are then read in degrees, and changed to radians by a call to CORRAD. The various paths from transmitter to target and then defined (RLA(ITCT), RLO(ITCT)), and a call to subroutine GEOPTS defines the incremental path coordinates. The call to subroutine GEODI returns the total distance from transmitter to target (SBKMS) by a Sodano calculation.[2] This value is later used to calculate the primary wave delay. With the incremental geographic coordinates known along the geodesic, the call to subroutine GEORET returns the ground electrical properties of elevation and impedance through common blocks/GROUND/and/SDRDI/for use in subroutine INEQ. Subroutine INEQ determines the secondary phase factor, TIMSAV. The time of arrival (TOA) for a given path (ITCT) is determined from the relationship:

$$\text{TOA (ITCT)} = \text{ENC} * \text{SBKMS} + \text{TIMSAV} \tag{2}$$

where ENC = ground refractive index divided by the velocity of light. ITCT is then incremented for a new path between the next transmitter in the chain and the same receiver, and program control returns to statement 15 where RLA(ITCT) and RLO(ITCT) are redefined. The above process is then repeated until three TOA'S are calculated. The required time difference (TDI) then computed from the relation:

$$TDI = EDI + TOA(2) - TOA(1) \tag{3}$$

where

EDI = emission delay for slave 1.
TOA(2) = Time of arrival for slave 1 at receiver, TOAS1.
TOA(1) = Time of arrival for Master at receiver, TOAM.

When the calculations are performed for airborne locations, the secondary phase factor contains an altitude correction derived in subroutine INEQ and defined as ALTTMSV.

Upon completion of the time difference calculation, control is transferred back to statement 5 in the program where the information for the next target is read in and the entire process repeated. The program exits when the step size (ADELS) on the target card is equal to or less than zero.

3.5.2 SUBROUTINE CORRAD

Subroutine CORRAD transforms degrees into radians for a given latitude or longitude. The subroutine statement is SUBROUTINE CORRAD (RCOR, IDEG, IMIN, SEC, ID, IS, IERR) where:

RCOR = Location in radians.
IDEG = Location in degrees.
IMIN = Location in minutes.
SEC = Location in seconds.
ID = Character for north, south, east, or west.
IS = Latitude or longitude indicator.
IERR = Error code.

This subroutine is called from the driving program and returns radians to the driving program through the argument list. It is used to transform the input transmitter and receiver coordinates into radians.

3.5.3 SUBROUTINE GEOPTS

Given the distance S_p from point A to a point P on the geodesic between two prescribed points A, B, on the surface of a spheroidal earth, the FORTRAN subroutines GEOPTS returns the latitude ϕ_p, and longitude θ_p of P, and the forward azimuth ψ_p of the geodesic of P as shown in Figure 6.

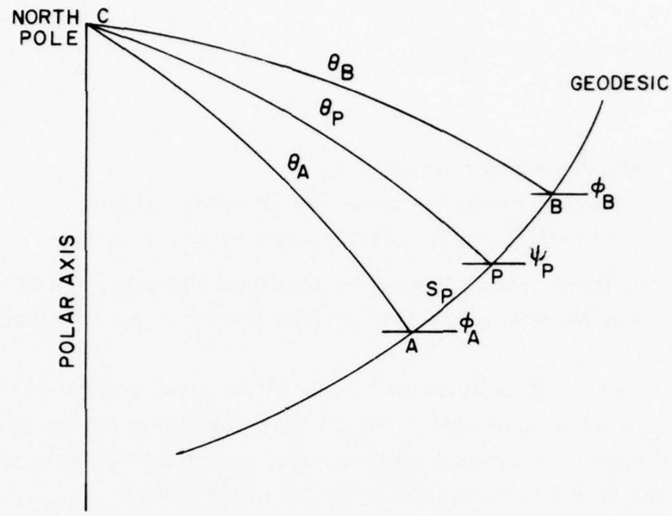


Figure 6. Geodesic Geometry

The derivation of the equations under the subroutine are described by Spies.[6] The subroutine statement is: SUBROUTINE GEOPTS (SMP, RLATP, RLONP, RAZP) where

SMP = S_p = distance (in meters) of P from A,

RLATP = ϕ_p = latitude (in radians) of P,

RLONP - θ_p = longitude (in radians) of P,

and

RAZP = ψ_p = forward azimuth (in radians) of geodesic of P.

Latitudes ϕ_A, ϕ_B and longitudes θ_A, θ_B of the prescribed end-points A, B are stored in the COMMON block PATH, along with certain output parameters. We

6. Spies, K.P. (1975) The Analytical Basis of Hufford's Computer Technique for Determining Topographic Profiles, Institute for Telecommunication Services Memo dated Nov. 4, 1975.

have the statement: COMMON/PATH/PLATA, RLONA, RLATB, RLONB, RAZA, RAZB, SMB, where

RLATA = ϕ_A = latitude (in radians) of A,

RLONA = θ_A = longitude (in radians) of A,

RLATB = ϕ_B = latitude (in radians) of B,

RLONB = θ_B = longitude (in radians) of B,

RAZA = ψ_A = forward azimuth (in radians) of geodesic of A,

RAZB = ψ_B = forward azimuth (in radians) of geodesic of B,

and

SMB = S_B = length (in meters) of geodesic from A to B.

North latitudes and east longitudes are positive, whereas south latitudes and west longitudes are negative. Positive azimuths are measured clockwise from north.

Parameters which specify relevant dimensions of the spheroidal earth are defined in the DATA statement: DATA ASPH, ESQ, CESQ/6378206.4, 0.006768658, 0.993231342/ where

ASPH = a = length (in meters) of semimajor axis of spheroid,

ESQ = $e^2 = (a^2 - b^2)/a^2$ (where b is length of semiminor axis of spheroid),

CESQ = $1 - e^2$,

and the numerical values correspond to the Clarke spheroid of 1866. Any other desired spheroid can be used in GEOPTS simply by replacing the above DATA statement by one containing the appropriate numerical values. For example, for the International spheroid, one would use:

DATA ASPH, ESQ, CESQ/6378388.0, 0.006722670, 0.993277330/.

As we shall see in the following section, the quantities ϕ_p, θ_p, ψ_p, are approximated by esculatory cubic polynomial functions of the distance S_p of P from A; for a fixed spheroid, the coefficients in these polynomials depend only on the location of the end-points A, B. To achieve computational efficiency in situations where ϕ_p, θ_p, ψ_p are to be calculated for more than one point P on the geodesic between a fixed pair of end points, GEOPTS is provided with a second entry point PCOORD (by including the ENTRY statement ENTRY PCOORD), whereby the subroutine skips the coefficient calculations and proceeds directly to the evaluation of the polynomial expressions for ϕ_p, θ_p, ψ_p. For a given pair of end-points A and B, the initial call to the subroutine must use the main entry point GEOPTS: that is, the calling program reference must be

CALL GEOPTS (etc., etc., ...).

Since the polynomial coefficients have already been evaluated, subsequent calls (for that path) should then use the entry point PCOORD: that is, the calling program reference should be:

CALL PCOORD (etc., etc., ...).

It is thus evident the subroutine GEOPTS is particularly efficient in those cases where ϕ_p, θ_p, ψ_p are desired for several to many points P on a single geodesic.

3.5.4 SUBROUTINE GEORET

This subroutine is called from the driving program LORANCO. Subroutine GEORET obtains from the data through GETELV the values of elevation and impedance which occur at intervals along a geodesic specified by arrays LAT and LON. The path elevation and impedance data are transmitted to the driving program via blocked common statements.

The subroutine statement is SUBROUTINE GEORET (LAT, LONG, DSKM, NP) where:

LAT = Array of latitudes along a geodesic path.

LON = Array of longitudes along a geodesic path.

DSKM = Distance along geodesic path in KM.

NP = Number of points along the geodesic path.

To obtain elevations and impedances from the data base, GEORET steps through NP latitude and longitude points calling GETELV each time. Amplitude and phase data are returned, converted to real and imaginary values, and stored in arrays DR and DI. Elevation information is stored in array Z. Subroutine GEORET then calls subroutine INT to obtain the first and second derivative of the elevation data.

Support information required by GEORET and supplied by common blocks include:

(1) Common/GROUND/
(2) Common/INDUCT/
(3) Common/TE/
(4) Common/SDRDI/
(5) Common/CITCT/

By use of common blocks, certain of the information can be directed to the double integral subroutine (INEQ) and its support subroutines, while other control data are transmitted to the driving section of the overall program.

3.5.5 SUBROUTINE GETELV

This subroutine is called from subroutine GEORET. Subroutine GETELV indexes the input latitude and longitude coordinates for subroutine UNPACK to obtain the proper ground electrical properties off the disc.

The subroutine statement is SUBROUTINE GETELV (LAT, LONG, AMP, FAZ, ELVTN) where:

LAT, LON - The input coordinate of each incremented point along the geodesic path.

AMP, FAZ, ELVTN = The output complex impedance (AMP, FAZ) and elevation (ELVTN) from the disc data base.

Two indices are selected of adjacent longitude records, NPACK, which encompass the input coordinate. These two records are separated by 30 in. in longitude and cover a latitude of one degree or 120 points. The elevation data, ELVTN, is obtained by a linear interpolation within the 30 in. square surrounding the desired coordinate, and is stored in eleven bits of the available thirty bit word. The impedance data of the southwest corner of the 30-in. square is used to represent the entire square. No impedance interpolation is required. For the complex impedance data, eight bits are used for the magnitude and eleven bits for the phase.

The impedance and elevation data are returned to subroutine GEORET through the argument list.

3.5.6 SUBROUTINE SETUP

This subroutine provides the spheroidal data for subroutine GEODI. Inputs are the semimajor (AO) and semiminor (BO) axis of the earth in meters. The spheroidal flattening (FL) and eccentricity square (ESQ) are calculated. The spheroidal constants for various ellipsoids are:

Ellipsoid	Semimajor	Semiminor
International	6378388.0	6356911.9
Clarke 1866	6378206.4	6356583.8
Clarke 1880	6378249.1	6356514.9
Bessel	6377397.2	6356079.0

Subroutine SETUP is called from subroutine GEODI and returns the required constants through the COMMON/GENERAL/block.

3.5.7 SUBROUTINE INT

Subroutine INT is called by subroutine GEORET and calculates the first and second derivative of the elevation at each point along the geodesic path. The subroutine statement is SUBROUTINE INT (I, K) where:

I = Position in x and z arrays on which to center calculations.

K = Position in arrays Z and ZP to store calculated values.

An analytical expression for a parabolic fit to three data points closest and including the required point along the geodesic path is derived by Sheed.[7] Differentiating this curve twice yields the required first and second derivative of the elevation at the position on which the calculations are centered. The results are returned to GEORET for use in subroutine INEQ via Common/Ground/block.

3.5.8 SUBROUTINE UNPACK

Subroutine UNPACK takes each sixty word record from the data base and unpacks each word into two complete data points composed of elevation and impedance information. Each called record contains the data for 120 points. The original sixty-bit words when unpacked, allow thirty bits for each data point. These are allocated as follows:

Elevation in meters, ELEV, 11 bits, ±7 meters.

Magnitude of Impedance, AMP, 8 bits, ±2 ohms.

Phase of Impedance, FAZ, 11 bits, ±0.001 radians.

The information obtained is returned to GETELV.

For machines with 32 bit words, the UNPACK subroutine is altered so that one data point is obtained from each word.

3.5.9 SUBROUTINE INEQ

Subroutine INEQ is called from the driving program, LORANCO, and returns the secondary phase factor or additional time correction. The LAT and LON in the calling statement are the array of path length latitudes, and longitudes respectively.

The following constants are first set for the operation of this program:

NUT = 0 or 1 depending on type of receiver antenna.

RAD = radius of earth = 6.36739×10^6 meters.

ALPHA = Vertical lapse factor = 0.85.

FREQ = Frequency = 0.1 MHz.

NSTART = Index of first distance at which the field is to be found as a function of altitude = 0.5 km.

ETA = Refractive index = 1.000338.

---

7. Sheed, F. (1962) Theory and Problems of Numerical Analysis, Shaum's Outline Series, McGraw Hill Co.

The attenuation function is calculated at discrete points as a function of geodesic distance.[8] The amplitude or modulus of the complex attenuation function is the field intensity and the phase or argument is the secondary phase correction in radians. To solve for this attenuation function, the boundary conditions must be known, then the solution can be extended step by step by numerical integration.

By assuming that the ground is smooth just in front of the transmitter, the first few points can be calculated with classical formulas.[9] For the remaining points, an integral equation approach is employed which allows one to introduce variations in ground elevation and variations in ground impedance relative to a classical spheroid. Special care must be exercised in the integration near the beginning and end of the integration path because the integrand approaches infinity at these points. A Gaussian quadrature integration formula[10] is employed in the area of such singularities, and Simpson's rule is used in the rest of the interval. The effective ground impedance which combines the elevation and impedance variations is obtained through a common statement from subroutine GROUND. After the calculations along the ground are completed for a given transmitter to receive path, a height gain loop[4, 11] is activated if the altitude input data is other than zero. Upon completion of the integration, the results are returned to the driving program through common block/DELAY/.

Input information initially obtained by the driving program from the stored disc data base is transmitted to subroutine INEQ via common blocks:

Common/GROUND/and Common/TE/

Transmitter-receiver path increment data enters via common blocks:

Common/SS/and Common/SDRDI/

Integral equation control of variables from the driving program into subroutine INEQ enter via:

Common/CITCT/

Calculations of secondary phase factors in units of microseconds are returned to the calling program via:

Common/DELAY/

---

8. Johler, J.R., and Horowitz, S. (1973) Propagation of LORAN-C Ground and Ionospheric Wave Pulses, Office of Telecommunications Report 73-20 (Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402).

9. Johler, J.R., Keller, W.J., and Walters, L.C. (1956) Phase of the Low Radiofrequency Ground Wave, NBS Circular 573 (Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402

10. Abramovitz, M. and Stegun, I. (1964) Handbook of Mathematical Functions — NBS — Applied Math Series 55, U.S. Government Printing Office.

11. Scott, R. (1966) Phase of the Height-Gain Function of the Low Frequency Ground Wave, Report 2900-156-T of Project Michigan (Willow Run Laboratories, Ann Arbor, Michigan 48106).

```
      PROGRAM LCRANCO(INPUT=128,OUTPUT=128,TAPE2)
      DIMENSION IDENTF(3)
      DIMENSION LAT(999),LON(999)
      DIMENSION DISTDUM(3),TIMDUM(3)
      DIMENSION TOA(3)
      DIMENSION DISTSOD(3)
      DIMENSION TPW(3)
      DIMENSION ALTTMSV(3,2)
      DIMENSION PATHL(3)
      DIMENSION LDUM(3,4),ADUM(3,4),RLA(3),RLO(3)
      COMMON/PATH/RLATA,RLONA,RLATB,RLONB,RAZA,RAZB,SBM
      COMMON/CITCT/ITCT,LTOP,ALTT MSV
      COMMON/DELAY/NP,DISTSAV,TIMSAV
      COMMON/ZOTA/ARRAY(40)
      NAMELIST/NAM1/TOA,TD1,TD2,ED1,ED2,DISTDUM,TIMDUM,DISTSOD
     1,TPW
      DATA  RTODEG/57.2957795130823/
      DATA ITCT/1/
      DATA ARRAY/100.,1.000338,40.,.1,5.,1.,3.,3000.,10.,0.,1000.,
     1      .85,0.,1.,26*0./
      C=.299792
      EN=1.000338
      ENC=EN/C
C      READ TRANSMITTER COORDINATES FOR MASTER, S1,S2.
      PRINT 101
      PRINT 13
  101 FORMAT(*1TRANSMITTER COORDINATES*)
      MISKOD=0
      DO 913 I=1,3
      READ 102,IDENTF,LDUM(I,1),LDUM(I,2),ADUM(I,1),ADUM(I,2),
     1          LDUM(I,3),LDUM(I,4),ADUM(I,3),ADUM(I,4),TOA(I)
      PRINT 103,IDENTF,LDUM(I,1),LDUM(I,2),ADUM(I,1),ADUM(I,2),
     1          LDUM(I,3),LDUM(I,4),ADUM(I,3),ADUM(I,4)
  102 FORMAT(2X,3A8,2(I5,I3,F7.3,A1),F12.1,F5.2,I3)
  103 FORMAT(2X,3A8,2(2X,I5,I3,F7.3,A1))
  104 FORMAT(*1RECEIVER COORDINATES*)
      CALL CORRAD(RLO(I),LDUM(I,3),LDUM(I,4),ADUM(I,3),ADUM(I,4),1,MISKO
     1D)
      CALL CORRAD(RLA(I),LDUM(I,1),LDUM(I,2),ADUM(I,1),ADUM(I,2),0,MISKO
     1D)
      IF(MISKOD.EQ.0) GO TO 913
      PRINT 23
      GO TO 10
  913 CONTINUE
      ED1=TOA(2)
      ED2=TOA(3)
    5 CONTINUE
      READ 102,IDENTF,LATDEGB,LATMINB,SECLATB,LATIDB,LONDEGB,LONMINB,
     1SECLONB,LONIDB,ALT,ADELS,NUT
      IF(ADELS.LE.0) GO TO 10
      PRINT 104
      PRINT 13
      PRINT 103,IDENTF,LATDEGB,LATMINB,SECLATB,LATIDB,LONDEGB,LONMINB,
     1SECLONB,LONIDB
      ARRAY(10)=ALT
      ARRAY(14)=NUT
      MISKOD=0
```

```
      CALL  CORRAD(RLATB,LATDEGB,LATMINB,SECLATB,LATIDB,0,MISKOD)
      CALL  CORRAD(RLONB,LONDEGB,LONMINB,SECLONB,LONIDB,1,MISKOD)
      IF(MISKOD.EQ.0) GO TO 15
   20 PRINT 23
   23 FORMAT(//11X, 3H***, 9X, 29HERROR IN END-POINT COORDINATE, 10X, 3H
     +***/11X,54H***  CALCULATIONS FOR THIS PATH HAVE BEEN DELETED  ***)
      GO TO 5
   15 CONTINUE
      RLATA=RLA(ITCT)
      RLONA=RLO(ITCT)
   13 FORMAT(12H0   LOCATION,22X,8HLATITUDE,9X,9HLONGITUDE/
     1 26X,2(5X,13H(DEG-MIN-SEC))/)
      LTOP=0
   25 CALL  GEOPTS(0.0,RLATP,RLONP,RAZP)
      CALL  GEODI(RLATA,RLONA,RAZAS,SBMS,RLATB,RLONB,RAZBS)
      SBKMS=SBMS*.001
      DAZA = RTODEG*RAZA
      DAZB = RTODEG*RAZB
      SBKM = SBM*1.0E-3
      DEL=SBKM-SBKMS
      PRINT 2,ITCT,SBKMS,SBKM,DEL
    2 FORMAT(*0GEODESIC DISTANCES, TRANSMITTER*I2* (POINT A) TO RECEIVER
     1 (POINT B)*/7X*SODANO*9X*HUFFORD*6X*DIFFERENCE*/3F15.5)
      NP = SBKM/ADELS+1.0
      DSKM = NP
      DSM = SBM/DSKM
      DSKM = DSM*1.0E-3
      PRINT 33,SBKM,DSKM,DAZA,DAZB
   33 FORMAT(//18X, 20HLENGTH OF GEODESIC =, F11.5, 11H KILOMETERS//18X,
     + 20HDISTANCE INCREMENT =, F11.5, 11H KILOMETERS///12X, 34HFORWARD
     +AZIMUTH OF GEODESIC AT A =, F12.6, 8H DEGREES//12X, 34HFORWARD AZI
     +MUTH OF GEODESIC AT B =, F12.6, 8H DEGREES)
      LAT(1)=LDUM(ITCT,1)*10000+LDUM(ITCT,2)*100+IFIX(ADUM(ITCT,1))
      LON(1)=LDUM(ITCT,3)*10000+LDUM(ITCT,4)*100+IFIX(ADUM(ITCT,3))
      NPP2=NP+2
      SPM = 0.0
      DO 100 IP=1,NPP2
      SPM = SPM+DSM
      CALL  PCOORD(SPM,RLATP,RLONP,RAZP)
      MISKOD = 0
      CALL  CORDMS(RLATP,LATDEGP,LATMINP,SECLATP,LATIDP,0,MISKOD)
      CALL  CORDMS(RLONP,LONDEGP,LONMINP,SECLONP,LONIDP,1,MISKOD)
      IF (MISKOD) 30,35,30
   30 PRINT 43,IP
   43 FORMAT(//2X, 62H***  ERROR DETECTED DURING CONVERSION OF COORDINAT
     +ES FOR POINT, I5, 5H  ***/2X, 2H**, 7X, 52HFROM RADIANS TO (ALPHAM
     +ERIC) DEGREES-MINUTES-SECONDS, 7X, 3H***/2X, 3H***, 66X, 3H***/2X,
     + 3H***, 7X, 52HFURTHER CALCULATIONS FOR THIS PATH HAVE BEEN DELETE
     +D, 7X, 3H***)
      GO TO 5
   35 CONTINUE
      LAT(IP+1 )=LATDEGP*10000+LATMINP*100+IFIX(SECLATP)
      LON(IP+1 )=LONDEGP*10000+LONMINP*100+IFIX(SECLONP)
  100 CONTINUE
      IP=NP+3
      CALL  GEORET(LAT,LON,DSKM,IP )
C       ADJUST NO OF PTS TO MATCH INEQ2E.
```

```
      NP=NP+1
      CALL INEQ2E(LAT,LON)
      TOA(ITCT)=ENC*SBKMS  +TIMSAV
      TPW(ITCT)=ENC*SBKMS
      DISTSOD(ITCT)=SBKMS
      PATHL(ITCT)=SBKMS
      DISTDUM(ITCT)=DISTSAV
      TIMDUM(ITCT)=TIMSAV
      ITCT=ITCT+1
      IF(ITCT.LT.4)GO TO 15$IF(LTOP.LT.1) PRINT 410
  410 FORMAT(*0  GROUNDWAVE TIME DELAY CALCULATION*)
      TD1=ED1+TOA(2)-TOA(1)
      TD2=ED2+TOA(3)-TOA(1)
      PRINT NAM1
      IF(LTOP.LT.1) GO TO 6
      DO 8 L=1,LTOP
      PRINT 7
    7 FORMAT(*0 ALTITUDE TIME DELAY CALCULATIONS*)
      DO 9 I=1,3
      TOA(I)=ENC*PATHL(I)+ALTTMSV(I,L)
    9 CONTINUE
      TD1=ED1+TOA(2)-TOA(1)
      TD2=ED2+TOA(3)-TOA(1)
      PRINT NAM1
    8 CONTINUE
    6 CONTINUE
      ITCT=1
      GO TO 5
   10 CONTINUE
      PRINT 11
   11 FORMAT(*0 END OF INPUT DATA.*)
      END
```

```
      SUBROUTINE GEORET(LAT,LON,DSKM,NP)
      DIMENSION LAT(999),LON(999)
      COMMON /GROUND/ZDUM, Z, ZPDUM, ZP, XDUM, X, R0, DEN, R, U, OMX, WA
     1VE, R2, ZPP, D, DDD1, DDD2
      COMMON/INDUCT/WAV
      COMMON/TE/TLA,TLON,RLAT,RLON,HLAT,HLON,       ZPP1,DINC1,TPI,NPTS
     1,ZMAZ,WTH
      COMMON /SDRDI/S(999), DR(999), DI(999), LLM
      DIMENSION ZDUM(6), Z(1009), ZPDUM(6), ZP(1009), XDUM(6), X(1009),
     1 ZPP1(1009)
      COMMON/CITCT/ ITCT
      COMPLEX D,DEN,DDD1,DDD2
      TPI=6.283185307
      TLA=LAT(1)
      TLON=LON(1)
      RLAT=LAT(NF)
      RLON=LON(NF)
      DINC1=DSKM
      NPTS=NP
      S(1)=0.
      X(1)=0.
      DO 1 I=1,NP
      AAMP=.0744$FAZ=.74109$Z(I)=2.
```

```
      CALL GETELV(LAT(I),LON(I),AAMP,FAZ,Z(I) )
      IF(I.GT.1) S(I)=S(I-1)+DSKM
      DR(I) = AAMP * COS(FAZ)
      DI(I) = AAMP * SIN(FAZ)
    1 CONTINUE
      LLM=NP
C      GENERATE INDEPENDENT PATH TERRAIN ELEVATIONS RELATIVE TO TRANSMIT
      Q=Z(1)
      Z(1)=0.
      DO 2 I=2,NP
C      FOR NOW, CONVERT DISTANCE TO METERS.
      S(I)=S(I)*1000.
      X(I)=S(I)
      Z(I)=Z(I)-Q
    2 CONTINUE
C      OBRAIN ELEVATION DERAVITIVES.
      NPM1=NP-1
      DO 3 L=2,NPM1
      CALL INT(L,L)
      ZPP1(L)=ZPP
    3 CONTINUE
      ZP(1)=0.
      ZPP1(1)=0.
      ZP(NP)=ZF(NP-1)
      ZPP1(NP)=ZPP1(NP-1)
      RETURN
      END



      SUBROUTINE CORRAD(RCOR,IDEG,IMIN,SEC,ID,IS,IERR)
      DIMENSION IDS(4)
      DATA (IDS=1HN,1HE,1HS,1HW)
      ISS = IS
      IF (ISS) 10,5,15
    5 IF (ID.EQ.IDS(1))  GO TO 25
      IF (ID.EQ.IDS(3))  GO TO 30
   10 IERR = 1
      RETURN
   15 IF (ISS-1) 20,20,10
   20 IF (ID.EQ.IDS(2))  GO TO 25
      IF (ID.EQ.IDS(4))  GO TO 30
      IERR = 1
      RETURN
   25 SIGN = 1.0
      GO TO 35
   30 SIGN = -1.0
   35 IF (IDEG-180) 40,40,10
   40 IF (IMIN-60) 45,10,10
   45 IF (SEC-60.0) 50,10,10
   50 RCOR = SIGN*FLOAT(IDEG)*(1.74532925199433E-2)+FLOAT(IMIN)*
     + (2.90888208665722E-4)+SEC*4.84813681109536E-6
      RETURN
      END
```

```
      SUBROUTINE GEOPTS(SMP,RLATP,RLONP,RAZP)
      COMMON/PATH/RLATA,RLONA,RLATB,RLONB,RAZA,RAZB,SMB
      DATA ASPH,ESQ,CESQ/6378388.000, 6.722670022E-3, 9.932773300E-1/
      DATA PID2, PI, TWOPI/1.57079632679490, 3.14159265358979,
     + 6.28318530717959/
      IF (ABS(RLATA)-PID2) 5,10,10
    5 IF (ABS(RLATB)-PID2) 15,10,10
   10 PRINT 3,RLATA,RLATB
    3 FORMAT(1H1, 89H    SUBROUTINE GEOPTS CALLED WITH END-POINT LATITUD
     +E OUT OF ACCEPTABLE RANGE (-PI/2,PI/2)//13X, 16HLATITUDE OF A  =,
     +F8.2, 29H RADIANS     LATITUDE OF B  =, F8.2, 8H RADIANS)
      PRINT 13
   13 FORMAT(33X, 28HPROGRAM EXECUTION TERMINATED)
      STOP
   15 IF (ABS(RLONA)-TWOPI) 20,20,25
   20 IF (ABS(RLONB)-TWOPI) 30,30,25
   25 PRINT 23,RLONA,RLONB
   23 FORMAT(1H1, 88H    SUBROUTINE GEOPTS CALLED WITH END-POINT LONGITU
     +DE OUT OF ACCEPTABLE RANGE (-2PI,2PI)//10X, 17HLONGITUDE OF A  =,
     +F9.2, 31H RADIANS      LONGITUDE OF B  =, F9.2, 8H RADIANS)
      PRINT 13
      STOP
   30 ALONR = RLONA
      BLONR = RLONB
      IF (ALONR+PI) 35,35,40
   35 ALONR = ALONR+TWOPI
   40 IF (BLONR+PI) 45,45,50
   45 BLONR = BLONR+TWOPI
   50 BLON = BLONR-ALONR
      IF (BLON+PI) 55,70,75
   55 IF (BLONR) 60,65,65
   60 BLONR = BLONR+TWOPI
      GO TO 95
   65 ALONR = ALONR-TWOPI
      GO TO 95
   70 PRINT 33
   33 FORMAT(1H1, 88H    GEODESIC PATH INCLUDES A GEOGRAPHIC POLE - SUBR
     +OUTINE GEOPTS CANNOT HANDLE THIS CASE)
      PRINT 13
      STOP
   75 IF (BLON-PI) 95,70,80
   80 IF (BLONR-PI) 90,90,85
   85 BLONR = BLONR-TWOPI
      GO TO 95
   90 ALONR = ALONR+TWOPI
   95 SMB = RLATA+RLATB
      HETA = SIN(0.5*SMB)
      HETA = (1.0-ESQ*HETA*HETA)/CESQ
      HLAT = ASPH/(HETA*SQRT(HETA*CESQ))
      BLON = BLONR-ALONR
      ALON = HETA*BLON
      Q = SIN(ALON)
      BLAT = COS(RLATB)
      ALAT = COS(RLATA)
      SAZA = Q*BLAT
      SAZB = Q*ALAT
      Q = SIN(0.5*ALON)
```

```
      Q = Q*Q
      AAZ = RLATB-RLATA
      CAZB = (1.0-Q)*SIN(AAZ)
      Q = Q*SIN(SMB)
      CAZA = CAZB+Q
      CAZB = CAZB-Q
      Q = SQRT(SAZA*SAZA+CAZA*CAZA)
      SMB = HLAT*ASIN(Q)
      SAZA = SAZA/Q
      CAZA = CAZA/Q
      SAZB = SAZB/Q
      CAZB = CAZB/Q
      RAZA = ATAN2(SAZA,CAZA)
      RAZB = ATAN2(SAZB,CAZB)
      ALON = RAZB-RAZA
      IF (ABS(ALON)-PI) 120,105,105
  105 IF (ALON) 110,110,115
  110 RAZB = RAZB+TWOPI
      GO TO 120
  115 RAZA = RAZA+TWOPI
  120 HLAT = CESQ+ESQ*ALAT*ALAT
      HLON = ASPH/SQRT(HLAT)
      HLAT = HLON*CESQ/HLAT
      HLON = HLON*ALAT
      CLAT = CAZA/HLAT
      CLON = SAZA/HLON
      HLAT = CESQ+ESQ*BLAT*BLAT
      HLON = ASPH/SQRT(HLAT)
      HLAT = HLON*CESQ/HLAT
      HLON = HLON*BLAT
      ALAT = CAZB/HLAT
      BLAT = (3.0*AAZ/SMB-ALAT-2.0*CLAT)/SMB
      ALON = SAZB/HLON
      BLON = (3.0*BLON/SMB-ALON-2.0*CLON)/SMB
      AAZ = 3.0*SMB
      ALAT = ((ALAT-CLAT)/SMB-2.0*BLAT)/AAZ
      ALON = ((ALON-CLON)/SMB-2.0*BLON)/AAZ
      AAZ = (RAZB-RAZA)/SMB
      ENTRY PCOORD
      RLATP = ((ALAT*SMP+BLAT)*SMP+CLAT)*SMP+RLATA
      RLONP = ((ALON*SMP+BLON)*SMP+CLON)*SMP+ALONR
      RAZP = AAZ*SMP+RAZA
      RETURN
      END


      SUBROUTINE CORDMS(RCOR,IDEG,IMIN,SEC,ID,IS,IERR)
      DIMENSION IDS(4)
      DATA (IDS=1HN,1HE,1HS,1HW)
      RANG = RCOR
      SEC = ABS(RANG)*206264.8062470964
      ISS = IS
      IF (ISS) 5,10,15
    5 IERR = 1
      RETURN
   10 IF (SEC-324000.0005) 25,5,5
   15 IF (ISS-1) 20,20,5
   20 IF (SEC-648000.0005) 25,5,5
```

```
25 IF (RANG) 30,35,35
30 ISI = 2
   GO TO 40
35 ISI = 0
40 IDEG = SEC/3600.0
   IMIN = SEC/60.0-60.0*FLOAT(IDEG)
   SEC = SEC-3600.0*FLOAT(IDEG)-60.0*FLOAT(IMIN)
   ISEC = SEC
   SEC = SEC-FLOAT(ISEC)
   IF (SEC-0.9995) 60,45,45
45 SEC = 0.0
   ISEC = ISEC+1
   IF (ISEC-60) 60,50,50
50 ISEC = 0
   IMIN = IMIN+1
   IF (IMIN-60) 60,55,55
55 IMIN = 0
   IDEG = IDEG+1
60 LDX = ISI+ISS+1
   BEC=FLOAT(ISEC)+SEC
   ID=IDS(LDX)
   RETURN
   END


    SUBROUTINE INEQ2E(LAT,LON)
     DIMENSION LAT(999),LON(999)
    COMMON /55/DISS(999 )
    COMMON /GROUND/ZDUM, Z, ZPDUM, ZP, XDUM, X, R0, DEN, R, U, OMX, WA
   1VE, R2, ZPP, D, DDD1, DDD2
    COMMON/INDUCT/WAV
    COMMON/TE/TLA,TLON,RLAT,RLON,HLAT,HLON,       ZPP1,DINC1,TPI,NPTS
   1,ZMAZ,WTH
    COMMON /ZOTA/ARRAY
    COMMON /SDRDI/S(999), DR(999), DI(999), LLM
    COMMON/CITCT/ITCT,LTOP,ALTT MSV
    COMMON/DELAY/NP,DISTSAV,TIMSAV
    DIMENSION ALTTMSV(3,2)
    DIMENSION E(999 , 2), F(999 , 2)
    DIMENSION ZZ(5), H(5)
    DIMENSION ZDUM(6), Z(1009), ZPDUM(6), ZP(1009), XDUM(6), X(1009),
   1TZER(1), T(1000), GW(5), GX(5), EW(9), EX(9), WG(5), XS(9), WE(9),
   2 W(1000),ARRAY(40),ZPP1(1009)
    COMPLEX D, DEN, DDD1, DDD2, FLEAF, F1, F2, F3, G1, G2, G3, SUM, TE
   1RM, W, WE, WG, WW, WX
    COMPLEX TS,FIND,FINDH
    EQUIVALENCE (ARRAY(1), FKHZ), (ARRAY(2), ETA), (ARRAY(3), DMAX), (
   1ARRAY(4), DINC), (ARRAY(8), ZMIN), (ARRAY(9), ZINC), (ARRAY(10), Z
   2MAX), (ARRAY(11), FLAT)
    DATA IARRA/0/
    DATA ((GX(K), K = 1, 5) = .02216356881, .1878315676, .4615973615,
   1.7483346284, .9484939262)
    DATA ((GW(K), K = 1, 5) = .5910484494, .5385334386, .438172725,
   1  .2989026983, .1333426886)
    DATA ((EX(K), K = 1, 4) = .1051402826, .3762245145, .6989480124,
   1 .9373342493)
```

```
      DATA ((EW(K), K = 1, 4) = .06568051989, .1960962655, .2525273456,
     1.1523625357)
      DATA (TZER(1)=1.0)
    6 FORMAT  (//E20.7,5X,*ARRAY(1) = FKHZ = FREQUENCY IN KHZ*/
     1  E20.7,5X,*ARRAY(2) = ETA = AIR INDEX OF REFRACTION AT GROUND*/
     2 E20.7,5X,*ARRAY(3) = DMAX = MAXIMUM DISTANCE BETWEEN TRANSMITTER*
     3 ,* AND RECEIVER IN KM*/
     4  E20.7,5X,*ARRAY(4) = DINC = DISTANCE INCREMENT IN KM*/
     5  E20.7,5X,*ARRAY(5) = NSTART = INDEX OF THE FIRST DISTANCE AT*,
     6  * WHICH THE FIELD IS TO BE FOUND AS A FUNCTION OF ALTITUDE*/
     7  E20.7,5X,*ARRAY(6) = NZINC = THE FIELD IS TO BE FOUND AS A *,
     8  *FUNCTION OF ALTITUDE EVERY NZINC INCREMENTS IN THE */
     9  40X,*DISTANCE ARRAY -- NZINC MUST NEVER BE LESS THAN 1*)
    7 FORMAT (E20.7,5X,*ARRAY(7) = INDEX OF THE LAST DISTANCE AT WHICH*,
     1 * THE FIELD IS FOUND AS A FUNCTION OF ALTITUDE*/
     2  E20.7,5X, *ARRAY(8) = ZMIN = THE MINIMUM ALTITUDE ABOVE THE *,
     3  *SURFACE AT WHICH THE FIELD IS FOUND, METERS*/
     4  E20.7,5X, *ARRAY(9) = ZINC = THE ALTITUDE INCREMENT, METERS*/
     5  E20.7,5X,*ARRAY(10) = ZMAX = THE MAXIMUM ALTITUDE, METERS*/
     6 E20.7,5X,*ARRAY(11) = FLAT = THE FACTOR USED IN THE FLAT EARTH *,
     7 *TEST, METERS*/
     8  E20.7,5X,*ARRAY(12) = ALPHA = VERTICAL LAPSE FACTOR*/
     9 E20.7,5X,*ARRAY(13) = EFFECTIVE EARTHS RADIUS*/
     A E20.7,5X,*ARRAY(14) = NUT*//)
      IF(IARRA.GT.1) GO TO 9999
      IARRA=2
C     SET CONSTANTS
      NUT=ARRAY(14)
      RAD = 6.36739E+6 / ARRAY(12)
      ARRAY(13) = RAD
      FREQ = FKHZ * 1.E-3
      NSTART = ARRAY(5)
      NZINC = ARRAY(6)
      IF (ARRAY(6) .LT. 1.) NZINC = 1
      NZERO=0
      T(NZERO)=1.0
      WAVE = 2.095844729E-2 * FREQ
      WAVE = WAVE * ETA
      WAV=WAVE
      AMICRO = 1.0 / (TPI * FREQ)
      TX = SQRT(FREQ * ETA) * .0408389549
      X(1) = 0.
      W(1) = 1.
 9999 CONTINUE
      DECTX=X(2)
      DMAX=X(NPTS)/1000.
      DINC=DECTX/1000.
      ARRAY(4)=DINC
      DINC1=DINC
      ARRAY(7) = DMAX / DINC + 1.
      NZEND = ARRAY(7)
      PRINT 6,(ARRAY(I),I=1,6)
      PRINT 7,(ARRAY(I),I=7,14)
      NOFLAT = 1
      NGO = 1
      D = CMPLX(DR(1), DI(1)) * SQRT(FREQ / 0.1)
      IMOST = DMAX / DINC + .01
      DELTX = 1000. * DINC
      T(1) = 1. / SQRT(DELTX)
```

```
31    DO 32 K = 1, 5
      X( - K) = DELTX * GX(K)
      CALL GROUND ( -K, 2, 0, 0)
32    WG(K) = FLEAF(WAVE, 0., 0., X( - K), D)
      I = 2
37    CONTINUE
      ZPP=ZPP1(I)
      T(I) = 1. / SQRT(X(I) + DELTX)
      OMX = OMCOS(X(I) / RAD)
      IGO = 2 - MOD(I, 2)
      IL = I - IGO - 1
      RD = SQRT(2. * RAD * (RAD + Z(I)) * OMX + Z(I) * * 2)
      GO TO (40, 45), NOFLAT
40    IF (I .LE. 4 .OR. (FLAT .GT. X(I))) 41, 45
41    W(I) = FLEAF(WAVE, 0., 0., X(I), D)
      GO TO 90
45    SUM = 0.
      NOFLAT = 2
48    DO 50 K = 1, 5
      CALL GEOM(I, - K, 1, 0.)
      TERM = WG(K) * CMPLX(COS(WAVE * R), - SIN(WAVE * R)) * DEN
50    SUM = (U * GW(K)) / SQRT(X(I) - X( - K)) * TERM + SUM
      SUM = 3. * T(1) * SUM
      KK = 1
      IF (IL .LT. 3) GO TO 100
      DO 60 K = 3, IL
      CALL GEOM(I, K, 1, 0.)
      TERM = U * T(K - 1) * W(K) * CMPLX(COS(WAVE * R), SIN( - WAVE * R)
     1) * DEN
      GO TO (53, 55), KK
53    SUM = 4. * T(I - K) * TERM + SUM
      KK = 2
      GO TO 60
55    SUM = 2. * T(I - K) * TERM + SUM
      KK = 1
60    CONTINUE
100   CONTINUE
      CALL GEOM(I, 2, 1, 0.)
      SUM = T(I - 2) * T(1) * U * W(2) * CMPLX(COS(WAVE * R), - SIN(WAVE
     1 * R)) * DEN + SUM
62    CALL GEOM(I, I - IGO, 1, 0.)
      F2 = U * T(IL) * W(I - IGO) * CMPLX(COS(WAVE * R), - SIN(WAVE * R)
     1) * DEN
      SUM = (SUM + T(IGO) * F2) * .333333333 * DELTX
      GO TO (65, 66), IGO
65    F1 = F2
      F2 = TERM
      GO TO 70
66    CALL GEOM(I, I - 1, 1, 0.)
      F1 = U * T(I - 2) * W(I - 1) * CMPLX(COS(WAVE * R), - SIN(WAVE * R
     1)) * DEN
      SUM = SUM + .08333333333 * DELTX * (5. * T(1) * F1 + 8. * T(2) * F
     12 - T(3) * TERM)
70    Q = TX / T(I - 1)
      RHO = 1. + ZP(I) * * 2
      RHO = ZPP / RHO
75    TERM = 1.2 / T(1) * T(I - 1) * CMPLX(Q, Q) * (D + CMPLX(0., - RHO
     1/ WAVE))
      WX = 1. - CMPLX(Q, Q) * (SUM + 2. / T(1) * (.466666667 * F1 - .066
     1666667 * F2)) / (1. + TERM)
```

```
 85  W(I) = WX
 90  DIST = .001 * X(I)
     CALL INDF(0.,X(I),NUT,FIND)
     TS=W(I)*FIND
     AMP=CABS(TS)
     PHA=CANG(TS)
     PHIC = - (PHA - WAVE * (R0 - X(I)))
     SEC = PHIC * AMICRO
     IF(I.NE.NP) GO TO 9876
     DISTSAV=DIST
     TIMSAV=SEC
9876 CONTINUE
     I = I + 1
     IF(I.LE.NPTS)  GO TO 37
150  NGO = 2
     IF (ZMAX.LE.0.)    RETURN
     ITOP = (NZEND - NSTART) / NZINC + 1
     IF (ITOP .GT. 1500) GO TO 280
     LTOP=(ZMAX-ZMIN)/ZINC+1.1
     IF(LTOP.GT.5) GO TO 280
     F2=(0.,0.)
     G2=(0.,0.)
     OLDR=0.
     IF (ITOP .LT. 1) GO TO 105
       NPALT=NP-NSTART+1
     DO 200 IS = 1, ITOP
     I = NSTART + (IS - 1) * NZINC
     II=I
     IF (I.GE.NZEND)    II=NZEND-1
     IGO = 2 - MOD(I, 2)
     DO 155 K = 1, 4
     X(1000 + K) = X(I) - DELTX * EX(K)
     XS(K) = X(1000 + K)
 155 CALL GROUND (1000+K,II,0,0)
     CALL CNEVKEN(X, W, IMOST, XS, WE, 4, 5, 1)
     IF (LTOP .LT. 1) GO TO 110
     DO 195 L = 1, LTOP
     H(L) = ZMIN + (L - 1) * ZINC
     IF (H(L) .LT. Z(I)) 156, 154
156  WW = 0.
     GO TO 193
154  ZZ(L) = RAD + H(L)
     R0 = SQRT(2. * RAD * ZZ(L) * OMCOS(X(I) / RAD) + H(L) * * 2)
     SUM = 0.
     DO 157 K = 1, 5
     CALL GEOM(I, - K, 2, H(L))
     TERM = WG(K) * CMPLX(COS(WAVE * R), - SIN(WAVE * R)) * DEN
157  SUM = U * GW(K) * TERM / SQRT(X(I) - X( - K)) + SUM
     SUM = 3. * T(1) * SUM
     CALL GEOM(I, 3, 2, H(L))
     F1 = T(1) * T(I - 2) * U * DEN * W(2)
     G1 = CMPLX(COS(WAVE * R), - SIN(WAVE * R))
     SUM = SUM + F1 * G1
     ILO = I - IGO
     KGO = 1
     IF (ILO .LT. 3) GO TO 115
     DO 182 K = 3, ILO
     F3 = F2
     G3 = G2
     F2 = F1
```

```
      G2 = G1
      CALL GEOM(I, K, 2, H(L))
      G1 = CMPLX(COS(WAVE * R), - SIN(WAVE * R))
      F1 = T(K - 1) * T(I - K) * U * DEN * W(K)
      DELTG = WAVE * (R - OLDR)
      GO TO (158, 159, 172), KGO
158   SUM = SUM + 4. * F1 * G1
      KGO = 2
      GO TO 182
159   SUM = SUM + 2. * F1 * G1
      KGO = 1
      IF (ABS(DELTG) .GT. .2) 170, 182
170   KGO = 3
      SUM = SUM - F1 * G1
      GO TO 182
172   SUM = SUM + CMPLX(0., 3. / DELTG) * (G1 * (F1 - .5 * (F1 * CMPLX(2
     1., 3. * DELTG) - 4. * F2 * CMPLX(1., DELTG) + F3 * CMPLX(2., DELTG
     2)) / DELTG * * 2) - G2 * (F2 - .5 * (F1 * CMPLX(2., DELTG) - 4. *
     3F2 + F3 * CMPLX(2., - DELTG)) / DELTG * * 2))
182   OLDR = R
115   CONTINUE
      GO TO (188, 183), IGO
183   F3 = F2
      G3 = G2
      F2 = F1
      G2 = G1
      CALL GEOM(I, I - 1, 2, H(L))
      G1 = CMPLX(COS(WAVE * R), - SIN(WAVE * R))
      F1 = T(I - 2) * T(1) * U * DEN * W(I - 1)
      DELTG = WAVE * (R - OLDR)
      IF (ABS(DELTG) .GT. .2) 184, 185
184   SUM = SUM + CMPLX(0., 3. / DELTG) * (G1 * (F1 - .5 * (F1 * CMPLX(2
     1., 3. * DELTG) - 4. * F2 * CMPLX(1., DELTG) + F3 * CMPLX(2., DELTG
     2)) / DELTG * * 2) - G2 * (F2 - .5 * (F1 * CMPLX(2., DELTG) - 4. *
     3F2 + F3 * CMPLX(2., - DELTG)) / DELTG * * 2))
      GO TO 188
185   SUM = SUM + 1.25 * F1 * G1 + 2. * F2 * G2 - .25 * F3 * G3
188   DO 190 K = 1, 4
      CALL GEOM(I, 1000 + K, 3, H(L))
      SUM = SUM + 3. * EW(K) * WE(K) * CMPLX(COS(WAVE * R), - SIN(WAVE *
     1 R)) * DEN * U / (SQRT(XS(K)) * T(1))
190   CONTINUE
      SUM = .33333333 * DELTX * SUM
      Q = TX / T(I - 1)
      WW = (1. - CMPLX(Q, Q) * SUM) * .5
193   CONTINUE
      CALL INDF (H(L),X(I),NUT,FINDH)
      TS=WW*FINDH
      E(IS, L) = CABS(TS) * 1.257 * FREQ / R0
      F(IS, L) = - (CANG(TS) - WAVE * (R0 - X(I)))
195   CONTINUE
110   CONTINUE
      DISS(IS) = .001 * X(I)
200   CONTINUE
105   CONTINUE
      IF (LTOP .LT. 1) GO TO 120
      DO 250 L = 1, LTOP
      IF (ITOP .LT. 1) GO TO 125
      DO 240 IS = 1, ITOP
```

```
      FM = F(IS, L) * AMICRO
       IF(IS.NE.NFALT) GO TO 9877
      ALTTMSV(ITCT,L)=FM
9877  CONTINUE
240   CONTINUE
125   CONTINUE
250   CONTINUE
120   CONTINUE
280   CONTINUE
      RETURN
      END


      SUBROUTINE GROUND(I, K, IGO, HH)
      COMMON /GROUND/ZDUM, Z, ZPDUM, ZP, XDUM, X, R0, DEN, R, U, OMX, WA
     1VE, R2, ZPP, DELTA, DELTA1, DELTA2
      COMMON /ZOTA/ARRAY
      COMMON /SDRDI/S(999), DR(999), DI(999), LLM
      DIMENSION ZDUM(6), Z(1009), ZPDUM(6), ZP(1009), XDUM(6), X(1009),
     1ARRAY(40)
      COMPLEX DELTA, DEN, DELTA1, DELTA2
      EQUIVALENCE (ARRAY(1), FKHZ), (ARRAY(13), A)
      CALL INT (K,I)
      FREQ = FKHZ * 1.E-3
      RETURN
      ENTRY GEOM
      HIT = Z(I) + HH
      IF (I .EQ. K) GO TO 20
      T = (X(I) - X(K)) / A
      GS = A + Z(K)
      GX = A + HIT
      CT = COS(T)
      ST = SIN(T)
      OT = OMCOS(T)
      R2 = SQRT(2. * GS * GX * OT + (HIT - Z(K)) * * 2)
      R1 = SQRT(2. * A * GS * OMCOS(X(K) / A) + Z(K) * * 2)
      R = R1 + R2 - R0
      U = X(K) * R0 * SQRT(1 + ZP(K) * * 2) / (R1 * R2 * X(I))
      IF (IGO .LT. 3) U = U * (X(I) - X(K))
      PD = (A * OT + Z(K) - HIT * CT + GX * ZP(K) * A * ST / GS) / R2
      XK = X(K)
      IF (XK .GE. S(LLM)) GO TO 12
      LLL = 1
      IF (XK .LT. S(1)) GO TO 10
      LLMO = LLM - 1
      IF (LLMO .LT. 1) GO TO 100
      DO 13 LLL = 1, LLMO
      IF (XK .LT. S(LLL + 1) .AND. XK .GE. S(LLL)) GO TO 10
 13   CONTINUE
100   CONTINUE
      GO TO 21
 10   DELTA = CMPLX(DR(LLL), DI(LLL)) * SQRT(FREQ / 0.1)
      GO TO 21
 12   DELTA = CMPLX(DR(LLM), DI(LLM)) * SQRT(FREQ / 0.1)
 21   CONTINUE
      DELTA2 = CMPLX(1., - 1. / (WAVE * R2)) * PD
      DELTA1 = DELTA - DELTA2
 19   DEN = DELTA2 + DELTA
      RETURN
```

```
20  U = 0.
    R2 = GX - A - Z(I)
    RETURN
    END
```

```
    SUBROUTINE GETELV(LATX,LONX,AMP,FAZ,ELVTN)
    COMMON/GETELV/L(120),IMP(120)
    COMMON/UNPACK/NPACK(60)$COMMON/FLAGS/IXPT,NFLAG
    DIMENSION MSINDX(5800)
    DATA KPS1/1/
    INDDIM=5800
    IF(KPS1.EQ.1)CALL OPENMS(2,MSINDX,INDDIM,0)$KPS1=0
    M11=3777B$M8=377B
    IF(LATX.LT.480000  .OR.  LATX.GE.540000) GO TO 190
    IF(LONX.LT.060000  .OR.  LONX.GE.140000) GO TO 190
    NFLAG=0$LATSEC=MOD(LATX,100)$LONSEC=MOD(LONX,100)
    LATDM=LATX-LATSEC$LONDM=LONX-LONSEC$IF(LATSEC.GE.30)GOTO40
    LATS=LATDM$LATN=LATDM+30$GO TO 60
 40 LATS=LATDM+30$LATN=LATDM+100$LATM=MOD(LATDM,10000)
    IF(LATM.EQ.5900) LATN=LATDM+4100
 60 IF(LONSEC.GE.30)GOTO80$LONW=LONDM$LONE=LONDM+30$GOTO100
 80 LONW=LONDM+30$LONE=LONDM+100
    LONM=MOD(LONDM,10000)
    IF(LONM.EQ.5900) LONE=LONDM+4100
100 CONTINUE
    INDNE=INDEX(LATN,LONE)
    IF(NFLAG.EQ.1) GO TO 120
     CALL READMS(2,NPACK,60,INDNE)
    CALL UNPACK
    INE=L(IXPT)
    IF(MOD(LATN,10000).EQ.0.OR.MOD(LONE,20000).EQ.0)GOTO120
    ISE=L(IXPT-1)
    CALL READMS(2,NPACK,60,INDNE-1)$CALL UNPACK
    INW=L(IXPT)$ISW=L(IXPT-1)$GO TO 180
120 CONTINUE
    INDNW=INDEX(LATN,LONW)
    IF(NFLAG.EQ.1) GO TO 140
    CALL READMS(2,NPACK,60,INDNW)
    CALL UNPACK
    INW=L(IXPT)
140 CONTINUE
    INDSE=INDEX(LATS,LONE)
    IF(NFLAG.EQ.1) GO TO 160
    CALL READMS(2,NPACK,60,INDSE)
    CALL UNPACK
    ISE=L(IXPT)
160 CONTINUE
    INDSW=INDEX(LATS,LONW)
    IF(NFLAG.EQ.1) GO TO 180
    CALL READMS(2,NPACK,60,INDSW)
    CALL UNPACK
    ISW=L(IXPT)
180 CONTINUE
    AMP=FLOAT(SHIFT(IMP(IXPT),-11).A.M8)*.001
     FAZ=FLOAT(IMP(IXPT).A.M11)*.001
    IF(NFLAG.EQ.1) PRINT 3100
```

```
      ELEAST=FLOAT(ISE)+FLOAT((INE-ISE)*(LATSEC-(LATSEC/30)*30))/30.
      ELWEST=FLOAT(ISW)+FLOAT((INW-ISW)*(LATSEC-(LATSEC/30)*30))/30.
      ELVTN=ELWEST+(ELEAST-ELWEST)*FLOAT(LONSEC-(LONSEC/30)*30)/30.
      RETURN
  190 PRINT 2070
      RETURN
 2070 FORMAT(* *,4H****,*LAT,LON REQUESTED ARE OUTSIDE MAP REGION - NO E
     1LEVATION RETURNED*,//)
 3100 FORMAT(* *,4H****,*INDEX REQUESTED EXCEEDS SCANLINES GENERATED, TH
     1US PREVIOUS ELEVATIONS WILL BE USED*)
      END


      FUNCTION INDEX(LAT,LON)
      COMMON/FLAGS/IXPT,NFLAG
      LATS=MOD(LAT,100)
      LATM=MOD(LAT,10000)-LATS
      LATD=(LAT-LATM-LATS)/10000
      LATM=LATM/100
      LONS=MOD(LON,100)
      LONM=MOD(LON,10000)-LONS
      LOND=(LON-LONM-LONS)/10000
      LONM=LONM/100
      ISUBLN=(LOND*5)
      ISUBLN=(MOD(ISUBLN,10)/5)*120
      ILOND=((LOND-6)/2)*1440+ISUBLN+LONM*2+LONS/30+1
      INDEX=ILOND+(LATD-48)*240
      IXPT=LATM*2+LATS/30+1
      IF(INDEX.GT.5760)NFLAG=1
      RETURN
      END


      SUBROUTINE UNPACK
      COMMON/GETELV/L(120),IMP(120)
       COMMON/UNPACK/NPACK(60)
      DATA MSK1/3777B/,MSK2/17777777B/
      DO 20 MA=1,60$M=2*(MA-1)+1
      L(M)=AND(MSK1,SHIFT(NPACK(MA),-49))*8
      IMP(M)=AND(MSK2,SHIFT(NPACK(MA),30))
      L(M+1)=AND(MSK1,SHIFT(NPACK(MA),-19))*8
      IF(L(M).GT.2000)L(M)=96$IF(L(M+1).GT.2000)L(M+1)=96
      IMP(M+1)=AND(MSK2,NPACK(MA))
   20 CONTINUE$END
```

```
      SUBROUTINE GEODI(B1, AL1, AZ12, S, B2, AL2, AZ21)
C-----CALCULATES INVERSE COMPUTATION FORM (SODANO, 1965)
C-----B1,AL1,B2,AL2 ARE GIVEN DATA
C-----AZ12,S,   AZ21 ARE CALCULATED DATA
C-----B1 = GEODETIC LATITUDE, RADIANS, OF FIRST POINT
C-----AL1 = GEODETIC LONGITUDE, RADIANS, OF FIRST POINT
C-----B2 = GEODETIC LATITUDE, RADIANS, OF SECOND POINT
C-----AL2 = GEODETIC LONGITUDE, RADIANS, OF SECOND POINT
C-----AZ12 = FORWARD AZIMUTH, RADIANS, FROM FIRST POINT
C-----AZ21 = BACK AZIMUTH,RADIANS, FROM SECOND POINT
C-----S = DISTANCE (LENGTH OF GEODETIC LINE) BETWEEN POINTS, METERS
C-----SOUTHERN LATITUDES AND WESTERN LONGITUDES ARE NEGATIVE
      COMMON /GENERAL/TPI, PI, PI2, PI4, A0, B0, FL, ESQ, IFLAG
      DATA (IFLAG = 1)
      IF (IFLAG .EQ. 1)   CALL SETUP
      IF (ABS(B1) .GT. PI4) GO TO 1
      TBET1 = TAN(B1) * (1. - FL)
      BET1 = ATAN(TBET1)
      GO TO 2
    1 CBET1 = 1.0 / TAN(B1) / (1. - FL)
      BET1 = ATAN(1. / CBET1)
    2 CONTINUE
      ALL1 = AL2 - AL1
      IF (AL2 - AL1 .EQ. 0.) 8, 9
    8 ALL2 = 0.
      GO TO 3
    9 CONTINUE
      ALL2 = AL2 - AL1 - SIGN (TPI, AL2 - AL1)
    3 IF (ABS(ALL1) .GT. ABS(ALL2)) 5, 6
    5 ALL = ALL2
      GO TO 7
    6 ALL = ALL1
    7 CONTINUE
   12 IF (ABS(ALL) .EQ. 0. .OR. ABS(ALL) .EQ. PI .OR. ABS(ALL) .EQ. TPI)
     1 10, 11
   10 ALL = ABS(ALL)
   11 CONTINUE
      IF (ABS(B2) .GT. PI4) GO TO 16
      TBET2 = TAN(B2) * (1. - FL)
      BET2 = ATAN(TBET2)
      GO TO 17
   16 CBET2 = 1.0 / TAN(B2) / (1. - FL)
      BET2 = ATAN(1. / CBET2)
   17 CONTINUE
      CBET1 = COS(BET1)
      SBET1 = SIN(BET1)
      CBET2 = COS(BET2)
      SBET2 = SIN(BET2)
      A = SBET1 * SBET2
      B = CBET1 * CBET2
      AB = SBET1 * CBET2
      BA = SBET2 * CBET1
      COSL = COS(ALL)
      SINL = SIN(ALL)
      CPHI = A + B * COSL
      SPHI = SQRT ((SINL * CBET2)**2 + (BA - AB * COSL)**2)
      IF (SPHI .GE. 0. .AND. CPHI .GE. 0.) 20, 21
```

```
20    PHI = ASIN(SPHI)
      IF (SPHI .GT. CPHI) PHI = ACOS(CPHI)
      GO TO 30
21    IF (SPHI .GE. 0. .AND. CPHI .LE. 0.) 22, 23
22    PHI = PI - ASIN(SPHI)
      IF (SPHI .GT. ABS(CPHI)) PHI = ACOS(CPHI)
      GO TO 30
23    IF (SPHI .LT. 0. .AND. CPHI .LT. 0.) 24, 25
24    PHI = PI - ASIN(SPHI)
      IF (ABS(SPHI) .GT. ABS(CPHI)) PHI = TPI - ACOS(CPHI)
      GO TO 30
25    IF (SPHI .LT. 0. .AND. CPHI .GE. 0.) 26, 27
26    PHI = TPI + ASIN(SPHI)
      IF (ABS(SPHI) .GT. CPHI) PHI = TPI - ACOS(CPHI)
      GO TO 30
27    CALL EXIT
30    CONTINUE
      C = B * SINL / SPHI
      FL2 = FL * FL
      CON1 = FL + FL2
      CON2 = 0.5 * FL2
      CON3 = SPHI * CPHI
      CON4 = PHI**2 / SPHI
      CON5 = CON4 * CPHI
      EM = 1. - C * * 2
      RATIO1 = (1. + CON1) * PHI + A * (CON1 * SPHI - CON2 * CON4)
     1 + EM * (-0.5 * CON1 * (PHI + CON3) + CON2 * CON5) - A * A * CON2
     2 * CON3 + EM * EM * CON2 * (0.125 * (PHI + CON3) - CON5 - 0.25 *
     3 CON3 * CPHI**2) + A * EM * CON2 * (CON4 + CON3 * CPHI)
      S = RATIO1 * B0
      IF (S .LE. 1.E-4) S = 0.
      RATIO2 = CON1 * PHI - A * CON2 * (SPHI + 2. * CON4)
     1 + 0.5 * EM * CON2 * (-5.0 * PHI + CON3 + 4.0 * CON5)
      ALAM = RATIO2 * C + ALL
      SALAM = SIN(ALAM)
      CALAM = COS(ALAM)
      CTAZ12 = BA - CALAM * AB
      CTAZ21 = BA * CALAM - AB
      IF (AL1 - AL2 .EQ. 0.)    35, 39
35    AZ12 = 0.
      AZ21 = 0.
      GO TO 34
39    CTAZ12 = CTAZ12 / (SALAM * CBET2)
      IF (CTAZ12 .EQ. 0.) 54, 55
54    AZ12 = PI2
      GO TO 56
55    CONTINUE
      AZ12 = ATAN(1. / CTAZ12)
56    CONTINUE
      CTAZ21 = CTAZ21 / (SALAM * CBET1)
      IF (CTAZ21 .EQ. 0.) 57, 58
57    AZ21 = PI2
      GO TO 34
58    CONTINUE
      AZ21 = ATAN(1. / CTAZ21)
34    CONTINUE
      IF (ALL .GE. 0. .AND. CTAZ12 .GE. 0.) 40, 41
```

```
40   AZ12 = AZ12
     GO TO 50
41   IF (ALL .GE. 0. .AND. CTAZ12 .LT. 0.) 42, 43
42   AZ12 = PI + AZ12
     GO TO 50
43   IF (ALL .LT. 0. .AND. CTAZ12 .GE. 0.) 44, 45
44   AZ12 = PI + AZ12
     GO TO 50
45   AZ12 = TPI + AZ12
50   IF (ALL .GE. 0. .AND. CTAZ21 .GE. 0.) 46, 47
46   AZ21 = PI + AZ21
     GO TO 51
47   IF (ALL .GE. 0. .AND. CTAZ21 .LT. 0.) 48, 49
48   AZ21 = TPI + AZ21
     GO TO 51
49   IF (ALL .LT. 0. .AND. CTAZ21 .GE. 0.) 52, 53
52   AZ21 = AZ21
     GO TO 51
53   AZ21 = PI + AZ21
51   CONTINUE
     AZ12 = AMOD(AZ12, TPI)
     IF (AZ12 .LT. 0.) AZ12 = AZ12 + TPI
     AZ21 = AMOD(AZ21, TPI)
     IF (AZ21 .LT. 0.) AZ21 = AZ21 + TPI
     RETURN
     END


     SUBROUTINE SETUP
     COMMON /GENERAL/TPI, PI, PI2, PI4, A0, B0, FL, ESQ, IFLAG
     DATA (TPI=6.28318530717959)
     PI = TPI / 2.0
     PI2 = TPI / 4.0
     PI4 = TPI / 8.0
C-----A0=SEMIMAJOR AXIS OF CLARKE SPHEROID OF 1866, METERS
C-----B0=SEMIMINOR AXIS OF CLARKE SPHEROID OF 1866, METERS
     A0 = 6.3782064E+6
     B0 = 6.3565838E+6
C-----FL=SPHEROIDAL FLATTENING
C-----ESQ=SECOND ECCENTRICITY SQUARED
     FL = 1.0 - B0 / A0
     ESQ = (A0 * * 2 - B0 * * 2) / B0 * * 2
     IFLAG = 0
     RETURN
     END
```

```
      FUNCTION OMCOS(X)
C     OMOCOS(X)= 1 - COS(X)
      IF (ABS(X) .GT. .15) GO TO 40
      IF (X .EQ. 0.) GO TO 50
      S = X * X
      OMCOS = T = .5 * S
      R = 4.
   10 T = - T * S / (R * (R - 1.))
      OMCOS = CMCOS + T
      IF (T / OMCOS .LE. .5E-9) GO TO 51
      R = R + 2.
      GO TO 10
   40 OMCOS = 1. - COS(X)
      RETURN
   50 OMCOS = 0.
   51 RETURN
      END


      SUBROUTINE INT (I,K)
C     I = POSITICN IN X AND Z ARRAYS ON WHICH TO CENTER CALCULATIONS
C     K = POSITION IN ARRAYS Z AND ZP TO STORE CALCULATED VALUES
      COMMON /GROUND/ ZDUM,Z,ZPDUM,ZP,XDUM,X,R0,DEN,R,U,OMX,WAVE,R2,ZPP
     1,D,DDD1,DDD2
      COMPLEX DEN,D,DDD1,DDD2
      DIMENSION ZDUM(6),Z(1009),ZPDUM(6),ZP(1009),XDUM(6),X(1009)
      IMO=I-1
      IPO=I+1
      C=((Z(IPO)-Z(IMO))-(Z(I)-Z(IMO))*(X(IPO)-X(IMO))/(X(I)-X(IMO)))/
     1 ((X(IPO)*X(IPO)-X(IMO)*X(IMO))-(X(I)*X(I)-X(IMO)*X(IMO))*
     2 (X(IPO)-X(IMO))/(X(I)-X(IMO)))
      B=((Z(I)-Z(IMO))-C*(X(I)*X(I)-X(IMO)*X(IMO)))/(X(I)-X(IMO))
      A=Z(I)-X(I)*(B+C*X(I))
      Z(K)=A+X(K)*(B+C*X(K))
      ZP(K)=B+2.*C*X(K)
      ZPP=2.*C
      RETURN
      END


      SUBROUTINE CNEVKEN(A, FA, NA, X, FX, NX, NPT, KASE)
      COMPLEX FA, FX, FUNCT, POLY
      DIMENSION A(NA), FA(NA), X(NX), FX(NX), FUNCT(15), ABSC(15), DIF(1
     15), POLY(15)
    1 FORMAT (1H1/50H THE X VALUES ARE NOT ARRANGED IN ASCENDING ORDER./
     1/5X,4HI = E20.9,5X,7HX(I) = E20.9,5X,4HJ = E20.9,5X,7HA(J) = E20.9
     2///14X,1HX,20X,4HF(X))
    2 FORMAT (5X,2E20.9)
  200 FORMAT (1H1* THERE ARE NOT ENOUGH POINTS IN THE GIVEN ARRAY.*)
  205 FORMAT (1H1* THE GIVEN VALUE OF NPT WAS GREATER THAN NA.*/* THE SU
     1BROUTINE REDUCED NPT TO WITHIN THE LIMITS OF NA.*)
  210 FORMAT (1H1* NPT WAS INITIALLY GREATER THAN 15 - NPT =I5.*/* THE S
     1UBROUTINE SET NPT TO 15 AND CONTINUED.*)
      LOOP = 1
      IF (NPT - 15) 3, 3, 9
    9 PRINT 210, NPT
      NPT = 15
```

```
  3  IF (NPT - NA) 8, 8, 4
  4  NPT = NPT - 2
     IF (NPT - 1) 5, 5, 6
  5  PRINT 200
     CALL EXIT
  6  GO TO (7, 3), LOOP
  7  LOOP = 2
     PRINT 205
     GO TO 3
  8  NPT2 = NPT / 2
     GO TO (11, 12), KASE
 11  NSTART = 1
     MX = 1
     GO TO 16
 12  NSTART = NA - 3
     TEST = A(NSTART) + (A(NSTART + 1) - A(NSTART)) / 2
     IF (NX .LT. 1) GO TO 170
     DO 14 I = 1, NX
     IF (X(I) - TEST) 14, 14, 13
 13  MX = I
     GO TO 16
 14  CONTINUE
170  CONTINUE
 16  NSTOP = NA - 1
     IF (NX .LT. MX) GO TO 175
     DO 125 I = MX, NX
     IF (X(I) - A(1)) 135, 15, 10
 10  IF (X(I) - A(NA)) 25, 20, 130
 15  FX(I) = FA(1)
     GO TO 125
 20  FX(I) = FA(NA)
     GO TO 125
 25  IF (NSTOP .LT. NSTART) GO TO 180
     DO 85 J = NSTART, NSTOP
     IF (X(I) - A(J)) 32, 35, 30
 30  IF (X(I) - A(J + 1)) 45, 40, 85
 32  II = I - 1
     PRINT 1, I, X(I), J, A(J)
     IF (II) 34, 34, 33
 33  PRINT 2, (X(N), FX(N), N = 1, II)
 34  CALL EXIT
 35  FX(I) = FA(J)
     NSTART = J
     GO TO 125
 40  FX(I) = FA(J + 1)
     NSTART = J + 1
     GO TO 125
 45  NSTART = J
     IF (ABS(X(I) - A(J)) - ABS(X(I) - A(J + 1))) 50, 50, 55
 50  JJ = J
     GO TO 60
 55  JJ = J + 1
     GO TO 60
 85  CONTINUE
180  CONTINUE
 60  IF (JJ - NPT2) 135, 135, 70
 70  IF (JJ + NPT2 - NA) 80, 80, 130
 80  KK = JJ - NPT2 - 1
```

```
   90 IF (NPT .LT. 1) GO TO 185
      DO 95 K = 1, NPT
      KK = KK + 1
      FUNCT(K) = FA(KK)
      ABSC(K) = A(KK)
   95 DIF(K) = ABSC(K) - X(I)
  185 CONTINUE
      NTOP = NPT - 1
      LL = 1
  100 IF (NTOP .LT. 1) GO TO 190
      DO 105 L = 1, NTOP
      LLL = L + LL
  105 POLY(L) = (FUNCT(L) * DIF(LLL) - FUNCT(L + 1) * DIF(L)) / (ABSC(LL
     1L) - ABSC(L))
  190 CONTINUE
      IF (NTOP - 1) 120, 120, 110
  110 DO 115 M = 1, NTOP
  115 FUNCT(M) = POLY(M)
      NTOP = NTOP - 1
      LL = LL + 1
      GO TO 100
  130 INC = - 1
      KK = NA + 1
      GO TO 140
  135 INC = 1
      KK = 0
  140 IF (NPT .LT. 1) GO TO 215
      DO 145 K = 1, NPT
      KK = KK + INC
      FUNCT(K) = FA(KK)
      ABSC(K) = A(KK)
  145 DIF(K) = ABSC(K) - X(I)
  215 CONTINUE
      NTOP = NPT - 1
      LL = 1
  150 IF (NTOP .LT. 1) GO TO 220
      DO 155 L = 1, NTOP
      LLL = L + LL
  155 POLY(L) = (FUNCT(1) * DIF(LLL) - FUNCT(L + 1) * DIF(LL)) / (ABSC(L
     1LL) - ABSC(LL))
  220 CONTINUE
      IF (NTOP - 1) 120, 120, 160
  160 DO 165 M = 1, NTOP
  165 FUNCT(M) = POLY(M)
      NTOP = NTOP - 1
      LL = LL + 1
      GO TO 150
  120 FX(I) = POLY(1)
  125 CONTINUE
  175 CONTINUE
      RETURN
      END
```

```
      COMPLEX FUNCTION FLEAF(WAVE, H1, H2, XD, DELTAR)
      COMPLEX TEMP, Q, Z, Z2, ZZ, HWERF, WERFZ, WERF, ZWERF, DELTAR
      HD = H2 - H1
      TEMP = (0.7071067812, - 0.7071067812) * SQRT(.5 * WAVE)
      XD2 = SQRT(XD)
      Q = - TEMP * HD / XD2
      Z = TEMP * DELTAR * XD2 + Q
      ZZ = - Z
      ZI = AIMAG(ZZ)
      IF (ZI .LT. 0. .OR. (ABS(REAL(ZZ)) .LT. 6. .AND. ZI .LT. 6.)) GO T
     1O 10
      Z2 = ZZ * * 2
      HWERF = (Z2 - 2.) / (ZZ * (Z2 - 3.5))
      GO TO 12
C     WERF - COMPLEMENTARY ERROR FUNCTION
   10 WERFZ = WERF(ZZ)
      HWERF = ZZ - 0.5 * WERFZ / (ZZ * WERFZ + (0., - 0.56418958))
   12 ZWERF = Z + HWERF
      FLEAF = (Q * ZWERF - 0.5) / (Z * ZWERF - 0.5)
      RETURN
      END
```

```
      COMPLEX FUNCTION WERF(ZZZ)
      COMPLEX Z, ZZZ, ZV, V, Z2, C, W, S
      DIMENSION C(12), W(5, 4)
      EQUIVALENCE (S, C(12))
      DATA (C(1) = (.0, - .5641895835))
      DATA W/(1.,0.0),
     1 (3.678794411714423E-01,6.071577058413937E-01),
     2 (1.831563888873418E-02,3.400262170660662E-01),
     3 (1.234098040866788E-04,2.011573170376004E-01),
     4 (1.125351747192646E-07,1.459535899001528E-01),
     5 (4.275835761558070E-01,0.000000000000000E+00),
     6 (3.047442052569126E-01,2.082189382028316E-01),
     7 (1.402395813662779E-01,2.222134401798991E-01),
     8 (6.531777728904697E-02,1.739183154163490E-01),
     9 (3.628145648998864E-02,1.358389510006551E-01),
     A (2.553956763105058E-01,0.000000000000000E+00),
     B (2.184926152748907E-01,9.299780939260186E-02),
     C (1.479527595120158E-01,1.311797170842178E-01),
     D (9.271076642644332E-02,1.283169622282615E-01),
     E (5.968692961044590E-02,1.132100561244882E-01),
     F (1.790011511813930E-01,0.000000000000000E+00),
     G (1.642611363929861E-01,5.019713513524966E-02),
     H (1.307574696698522E-01,8.111265047745472E-02),
     I (9.640250558304439E-02,9.123632600421258E-02),
     J (6.979096164964750E-02,8.934000024036461E-02)/
      XX = REAL(ZZZ)
      YY = AIMAG(ZZZ)
      X = ABS(XX)
      Y = ABS(YY)
      Z = CMPLX(X, Y)
      LZ2 = 0
      IF (X .GE. 4.5 .OR. Y .GE. 3.5) GO TO 100
      I = X + .5
      J = Y + .5
      V = CMPLX(FLOAT(I), FLOAT(J))
```

```
      ZV = Z - V
      C(2) = W(I + 1, J + 1)
      AI = 0.
      DO 10 I = 3, 12
      AI = AI - .5
      C(I) = (V * C(I - 1) + C(I - 2)) / AI
   10 CONTINUE
      J = 12
      DO 11 I = 2, 11
      J = J - 1
   11 S = S * ZV + C(J)
   20 IF (YY .GE. 0.) GO TO 30
      IF (LZ2 .EQ. 0) Z2 = Z * Z
      S = 2. * CEXP( - Z2) - S
      IF (XX .GT. 0.) S = CONJG(S)
      GO TO 200
   30 IF (XX .LT. 0.) S = CONJG(S)
  200 WERF = S
      RETURN
  100 LZ2 = 1
      Z2 = Z * Z
      S = Z * ((0., 0.4613135279) / (Z2 - 0.1901635092) + (0., 0.0999921
     16168) / (Z2 - 1.7844927485) + (0., 0.0028838938748) / (Z2 - 5.5253
     24374379))
      GO TO 20
      END


      FUNCTION CANG(Z)
      COMPLEX Z
      DATA (PI=3.14159265358979),(PIHA=1.57079632679489)
      X=REAL(Z)
      Y=AIMAG(Z)
      IF (X) 20,30,10
   10 CANG=ATAN2(Y,X)
      RETURN
   20 IF (Y.NE.0.)   GO TO 10
      CANG=PI
      RETURN
   30 IF (Y.GT.0.)   GO TO 40
      IF (Y.LT.0.)   GO TO 50
      CANG=0.
      RETURN
   40 CANG=PIHA
      RETURN
   50 CANG=-PIHA
      RETURN
      END
```

```
      SUBROUTINE INDF (HH,XX,NUT,F)
C     THIS SUBROUTINE CALCULATES THE INDUCTION FIELDS FOR E SUB R AND
C     H SUB PHI
C     THESE INDUCTION FIELDS ARE FOR POSITIVE TIME FUNCTION
C     NUT = 1 GIVES INDUCTION FIELD FOR E SUB R (TAU BAR)
C     NUT = 0 GIVES INDUCTION FIELD FOR H SUB PHI (LOOP)
C     FZ = INDUCTION FIELD FOR E SUB R
C     FH = INDUCTION FIELD FOR H SUB PHI
      COMMON/INDUCT/WAVE
      DOUBLE THETAD,SINTH,COSTH,R,CONS
      COMPLEX GZ,FZ,FH,F
      A=6.36739 E6
      TPI=6.283185307
      C=2.997925E8
      IF(XX.LE.0.)3,4
    3 PRINT 5
    5 FORMAT (//* IN INDF, DISTANCE IS ZERO OR NEGATIVE, XX = *,E20.10)
      CALL EXIT
    4 THETAD=1.D0*XX/A
      SINTH=DSIN(THETAD)
      COSTH=DCOS(THETAD)
      R=A+HH
      CONS=A*R*SINTH**2
      D2=R*R+A*A-2.*A*R*COSTH
      IF(D2.GT.0.) GO TO 30
      DD=XX
      D2=XX*XX
      GO TO 40
   30 CONTINUE
      DD=SQRT(D2)
   40 CONTINUE
      D3=DD*D2
      D4=D2*D2
      FZR=-2.*COSTH/DD+3.*CONS/D3
      FZI=(WAVE*CONS+2.*COSTH/WAVE)/D2-3.*CONS/(D4*WAVE)
      FZ=CMPLX(FZR,FZI)
      GZ=CMPLX(0.,WAVE)
      FZ=FZ/GZ
      FHR=R*SINTH*DD/D3
      FHI=WAVE*R*SINTH*DD/D2
      FH=CMPLX(FHI,-FHR)
      C1=2.E-7*TPI*WAVE*C
      FH=FH/C1
      IF(NUT)1,2
    1 F=FZ
      RETURN
    2 F=FH
      RETURN
      END
```

3.7 Sample Test Run

A typical data output listing is illustrated below. The transmitter coordinates which are read in from the data card deck are printed for verification purposes. The geographic coordinates of the desired target or receiver are next listed. Array data used in the calculations are then printed. TOA is the travel time from transmitter to receiver in μsec and recorded in order of master, slave 1 and slave 2 to

target respectively. TD1 and TD2 are the final desired LORAN coordinates in μsec. ED1 and ED2 are the slave emission delays in μsec obtained from the transmitter card input data. Additional output data recorded as TOA above include:

- DISTDUM - Calculation of distance by Hufford's technique in KM.
- TIMDUM - Calculation of secondary phase factor in μsec.
- DISTSOD - Calculation of distance by Sodano technique in KM.
- TPW - Travel time of primary wave in μsec.

3.8 Output Listing

```
TRANSMITTER COORDINATES

  LOCATION                          LATITUDE          LONGITUDE
                                 (DEG-MIN-SEC)      (DEG-MIN-SEC)

 MASTER                          49 36 18.813N       7 19 38.276E
 SLAVE1                          53 39 13.867N       8 43 47.528E
 SLAVE2                          48 15 48.929N      11 37 49.263E

RECEIVER COORDINATES

  LOCATION                          LATITUDE          LONGITUDE
                                 (DEG-MIN-SEC)      (DEG-MIN-SEC)

 RECEIVER 5                      50 31 28.176N       8 37  4.102E

.10000000E+03      ARRAY(1) = FKHZ = FREQUENCY IN KHZ
.10003338E+01      ARRAY(2) = ETA = AIR INDEX OF REFRACTION AT GROUND
.33512978E+03      ARRAY(3) = DMAX = MAXIMUM DISTANCE BETWEEN TRANSMITTER
.99761245E+00      ARRAY(4) = DINC = DISTANCE INCREMENT IN KM
.50000000E+01      ARRAY(5) = NSTART = INDEX OF THE FIRST DISTANCE AT
.10000000E+01      ARRAY(6) = NZINC = THE FIELD IS TO BE FOUND AS A
                                 DISTANCE ARRAY -- NZINC MUST NEVER BE
.33700000E+03      ARRAY(7) = INDEX OF THE LAST DISTANCE AT WHICH THE
.30000000E+04      ARRAY(8) = ZMIN = THE MINIMUM ALTITUDE ABOVE THE
.10000000E+02      ARRAY(9) = ZINC = THE ALTITUDE INCREMENT, METERS
.30000000E+04      ARRAY(10) = ZMAX = THE MAXIMUM ALTITUDE, METERS
.10000000E+04      ARRAY(11) = FLAT = THE FACTOR USED IN THE FLAT EARTH
.85000000E+00      ARRAY(12) = ALPHA = VERTICAL LAPSE FACTOR
.74910047E+07      ARRAY(13) = EFFECTIVE EARTHS RADIUS
.10000000E+01      ARRAY(14) = NUT

$NAM1

TOA     = .45076707956884E+03, .11637208385365E+04, .11131957931839E+04,

TD1     = .13240373758969E+05,

TD2     = .29816538713615E+05,

ED1     = .1253742E+05,

ED2     = .2516411E+05,
```

```
DISTSUM = .1278157184542E+03, .7482066812882E+03, .3372025301828E+03,
TIMSUM  = .1525023417869E+01, .1671249018305E+01, .1421351331479E+01,
DISTSOD = .1279122027422E+03, .3482854402589E+03, .3331936655047E+03,
IPW     = .4598481055751E+03, .1162143626109E+04, .1111791792127.3E+16,
*END
END OF INPUT DATA.
```

4. DESCRIPTION OF PART II

4.1 Overview

The electrical properties of the earth are converted into a ground surface impedance in PART II of the program as shown in Figure 7. The data on the input tapes from PART I are loaded on to a disc so that at each index point information on terrain, geology, and soil is available. This information is employed to develop a 3-layer model of the ground as shown in Figure 8. The time of arrival of the LORAN pulse at the receiver is controlled by the electrical characteristics of this model through a parameter called the surface impedance. This parameter is the ratio of the horizontal components of the electrical to the magnetic field components assuming the ground is isotropic; the calculation of this parameter is described by Johler.[12, 13]

The development of the 3-layer ground model requires additional investigation, in particular, the constants to be used in the postulated second or saturated layer. The electrical characteristics of ground or other medium may be expressed by three constants, the relative permeability, the dielectric constant, and the conductivity. The relative permeability can normally be regarded as unity. From the digitized data from PART I, the conductivity of the top unsaturated ground or soil and basement rock can be determined.[14, 15, 16, 17, 18] Most soil maps also contain information on top soil depth which is included in the look-up table. The rock depth is assumed to be infinite since the 100 kHz wave does not penetrate below 300 meters. The conductivity and depth of the second layer are somewhat uncertain and various articles in the journal titled Geoexploration[19, 20] have been used to obtain information on second layer parameters as a function of the top layer. The values listed in Figure 8 are suitable for Europe. In Iran, for example, sigma 2 is larger than sigma 1. The dielectric constant has been selected as 15 over land and 80 over water. An empirical relationship between conductivity and dielectric constant has been developed by Hanle,[21] and may also be used in the

---

Due to the large number of references on this page, the references will not be footnoted. See References, page 84.

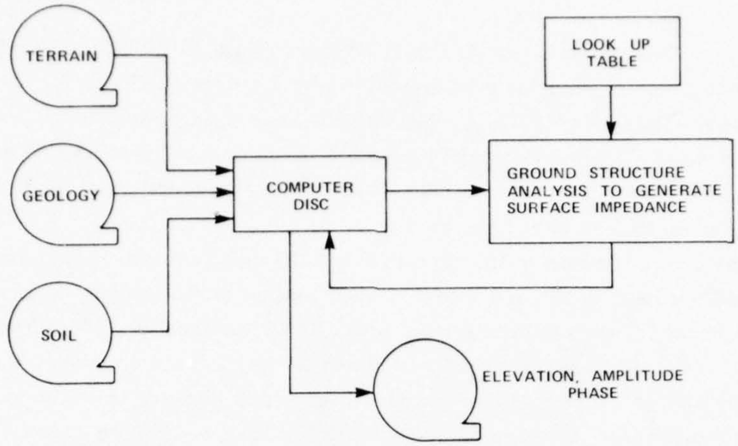


Figure 7. Data Base of Ground Electrical Properties - Part II

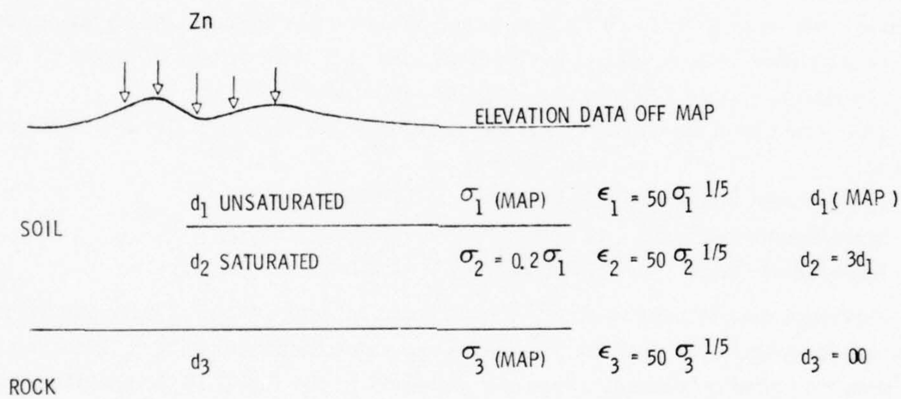


Figure 8. Ground Model for Surface Impedance Calculators

program. The effects of temperature changes, vegetation, buildings, and other objects on the surface of the earth have not been considered.

The input data to PART II is on a standard one-half inch, 400-foot magnetic tape containing information on elevation in meters, bedrock, and soil identification number. The input format for the data is as follows:

Elevation	- 11 bits	- ±7 meters
Blank	- 3 bits	
Rock Number	- 8 bits	- 256 types
Blank	- 2 bits	
Soil Number	- 6 bits	- 64 types

This totals to a 30-bit data word. Each CDC word contains 60 bits, therefore, two data words or points can be accommodated by a *single* CDC word. For 30-bit word computers (IBM 360, UNIVAC, Honeywell), one data point will be accommodated by each word allowing flexibility between machines. There are 60 words in each record or $2 \times 60 = 120$ data points. With a 30 arc second grid, each record covers one degree in latitude of the grid.

The data is recorded on to the tape in the following sequence: Starting at 6° E Long, 48° N Lat, in the southwest corner and go north for one degree in latitude (one record), then repeat for an equal line incremented in longitude, parallel and 30 arc seconds to the east of the first line. This is repeated until the last longitude line of the two degree standard longitude sectors is reached. For this particular problem, this would mean after $2 \times 60 \times 2 = 240$ records. The latitude is then incremented by one degree, returned to the original longitude, and the entire process repeated for another 240 records. At the east edge of the final latitude sector (53° to 54°), the pointer returns to 8° N Lat and begins another 1° latitude scan from 8° E to 10° E longitude. For the six degree latitude and eight degree longitude area coverage in Germany, there will be a total of $960 \times 6 = 5760$ records which contain $5760 \times 120 = 691,200$ data points.

The output data tape of PART II which is the input to PART III has the following format:

Elevation in meters	- ±7 meters	- 11 bits
Impedance Amplitude	- ±2 ohms	- 8 bits
Impedance Phase	- ±*.001 radians*	- 11 bits

The elevation data is passed directly from input to output tape. The output impedance is returned to the disc for storage in the same location as the original data and then recorded on magnetic tape for shipment to the PART III computation sites.

4.2 Subroutine Description

4.2.1 PROGRAM CONIMP

(a) Controls impedance program.

(b) Call subroutine SETUP.

(c) Reads and writes on disc soil, geology and elevation identification data off input tapes.

(d) Call subroutine REPK.

(e) Writes impedance data on disc and output tape.

4.2.2 SUBROUTINE SETUP

(a) Sets relative permeability of three layers to unity.

(b) Reads and stores ground conductivity tables from input cards into look-up file.

4.2.3 SUBROUTINE REPK

(a) Determine parameters for 3-layer ground surface impedance.

(1) Dielectric constant equals fifteen overland and eightly over

(2) Determine depth of first and second layer. Set third layer to infinity.

(3) Set sigma 2 as a function of sigma 1.

(b) Call subroutine GRIM for impedance calculation.

4.2.4 SUBROUTINE GRIM

(a) Calculates surface impedance for 3-layer ground model.

(b) Returns complex impedance (AMP, FAZ) to subroutine RFPK via argument list.

4.3 Resistivity Tables

For subroutine SETUP, a look-up table is required to convert type of soil and rock into resistivity values. The bibliography at the end of this section lists available sources for these values. Table 1 is a list of average values for the top or unsaturated soil. An approximate value is usually assigned in the map legend.

An inspection of the geological map of Europe reveals characteristic rock types which include anorthosite, granite gneiss, gabbro, Precambrian igneous, and metamorphic rock. The associated resistivities for various rock types are defined in Table 2.

Table 1. Ground Resistivity Values for Typical Soils

Terrain	Conductivity (mhos-meter)	Resistivity (meter-ohms)
Sea water	5	0.2
Fresh water	8×10^{-3}	123
Dry, sandy, flat coastal land	2×10^{-3}	500
Marshy, forested flat land	8×10^{-3}	123
Rich agricultural land, low hills	1×10^{-2}	1,000
Pastoral land, medium hills and forestation	5×10^{-3}	200
Rocky land, steep hills	2×10^{-3}	500
Mountainous	1×10^{-3}	1,000
Cities, residential areas	2×10^{-3}	500
Cities, industrial areas	1×10^{-4}	1,000

Table 2. Rock Types and Resistivities

Principle Geologic Age	Secondary Geologic Age	Principle Rock Type	Secondary Rock Types	Effective Resistivity Meter-ohms		
				min	max	mean
Precambrian		Gneisses, Granities, etc.		1, 000	14, 000	
Precambrian		Gneisses, Granites, etc.	Alluvium			200
Precambrian	Cambrian	Gneisses, Granites, etc.	Slate	400	7, 000	1, 500
Precambrian	Cambrian	Gneisses, Granities, etc.	Sandstone	1, 000	5, 000	2, 200
Precambrian	Cambrian & Ordovician	Gneisses, Granites, etc.	Quartzite & Limestone	1, 000	2, 000	1, 800
Precambrian	Cambrian & Ordovician	Gneisses, Granites, etc.	Limestone	65	180	140
Cambrian	Ordovician	Sandstone	Limestone & Shale	50	140	100
Cambrian	Ordovician Silurian Carboniferous	Sandstone	Limestone	150	480	300
Ordovician		Shale		55	70	65
Ordovician	Silurian Devonian	Sandstone	Limestone Shale	170	380	200
Ordovician	Silurian Devonian	Limestone	Shale			20

Table 2. Rock Types and Resistivities (Cont.)

Principle Geologic Age	Secondary Geologic Age	Principle Rock Type	Secondary Rock Types	Effective Resistivity Meter-ohms		
				min	max	mean
Ordovician	Carboniferous	Sandstone	Limestone & Shale	62	90	75
Cambrian	Ordovician	Sandstone	Limestone, Shale, Alluvium			6
Cambrian	Ordovician	Limestone	Shale	170	300	200
Cambrian	Ordovician Silurian	Sandstone	Limestone & Shale	38	180	130
Cambrian	Ordovician Carboniferous	Sandstone	Limestone & Shale	250	310	280
Cambrian	Ordovician Silurian Carboniferous	Sandstone	Limestone & Shale	350	605	480
Ordovician	Silurian, Devonian & Carboniferous	Sandstone	Limestone & Shale	71	105	100
Ordovician	Triassic	Sandstone	Limestone & Shale	170	270	205
Silurian	Devonian	Limestone		55	70	62
Silurian	Devonian	Limestone		62	84	72
Silurian	Devonian	Sandstone	Limestone & Shale	290	440	350
Devonian		Sandstone	Shale	200	400	290

Table 2. Rock Types and Resistivities (Cont.)

Principle Geologic Age	Secondary Geologic Age	Principle Rock Type	Secondary Rock Types	Effective Resistivity Meter-ohms		
				min	max	mean
Devonian	Carboniferous	Sandstone	Shale	19	34	29
Carboniferous		Sandstone	Shale			20
Carboniferous		Sandstone	Shale			10
Carboniferous		Sandstone	Limestone & Shale	20	40	25
Carboniferous		Limestone	Alluvium			28
Carboniferous		Limestone				2
Carboniferous	Triassic	Sandstone	Limestone	40	44	40
Carboniferous	Cretaceous Tertiary	Sand	Clay Loam			40
Carboniferous	Cretaceous Tertiary	Marl	Sand Clay	22	26	25
Carboniferous	Quarternary	Sand	Shale Alluvium	8	11	10
Triassic		Sandstone	Shale, Trap & Diobase	150	350	250
Cretaceous		Chalk				5.4
Cretaceous		Sandstone	Clay	10	15	13
Cretaceous	Tertiary	Sand	Clay	13	17	17

Table 2. Rock Types and Resistivities (Cont.)

Principle Geologic Age	Secondary Geologic Age	Principle Rock Type	Secondary Rock Types	Effective Resistivity Meter-ohms		
				min	max	mean
Cretaceous	Tertiary Quarternary	Sand	Clay			8
Cretaceous	Tertiary Quarternary	Marl	Sand Clay			10
Tertiary		Sand	Clay			31
Tertiary		Sand	Clay	4.5	7	6
Tertiary		Sand	Clay	5.5	6	6
Tertiary	Quarternary	Sand	Clay			15
Tertiary	Quarternary	Sand	Clay	18	20	19
Quarternary		Sand	Clay	4	8	5

4.4 Program Listing for Part II

PROGRAM CONIMP

```
      PROGRAM CONIMP(TAPE3,TAPE2,INPUT=128,OUTPUT=128)
      COMMON/THREE/F,PHI,SIGMA1,E21,AMU1,Z1,SIGMA2,E22,AMU2,Z2,SIGMA3,
     .E23,AMU3
       COMMON/CODE/SLCD(65,3),G0,GLCD(148)
      DIMENSION NUPK(60),NDXEQ(5800)
      DATA MSK6/77B/,MSKF15/77777000000000000000B/
      LNLM=960$LATLM=6
      LNLM=20$LATLM=1
      NDXSIZ=LNLM*LATLM+40
       CALL SETUP
      CALL OPENMS(2,NDXEQ,5800,0)
      LNLM=5760
      DO 100 ILN=1,LNLM
      NDXL=ILN
      BUFFERIN(3,1)(NUPK(1),NUPK(60))$IF(UNIT(3))10,20,30
   30 PRINT1,ILN
    1 FORMAT(*0  PARITY ERROR*,I9)
   10 CONTINUE
      IF(MOD(ILN,1440).LT.10)PRINT 9,ILN,NDXL,NUPK(1),NUPK(2)
      CALL REPK(NUPK,MSK6,MSKF15)
      IF(MOD(ILN,1440).LT.10)PRINT 99,NUPK(1),NUPK(2)
      IOVER=1
      IOVER=0
   70 CALL WRITMS(2,NUPK,60,NDXL,IOVER)
    9 FORMAT(*0 *,2I8,2O22)
   99 FORMAT(2O22)
  100 CONTINUE
   20 STOP$END
```

SUBROUTINE REPK

```
      SUBROUTINE REPK(NUPK,MSK6,MSKF15)
      COMMON/THREE/F,PHI,SIGMA1,E21,AMU1,Z1,SIGMA2,E22,AMU2,Z2,SIGMA3,
     .E23,AMU3
       COMMON/CODE/SLCD(65,3),G0,GLCD(148)
      DIMENSION NUPK(60),IAB(120)
      DATA MSK8/377B/
       MSK30=37770000003777000000B
      DO 100 I=1,60
      LLEV=NUPK(I).A.MSK30$LEV=SHIFT(LLEV,1)
      LS2=NUPK(I).A.MSK6$LG2=MSK8.A.SHIFT(NUPK(I),-6)
      LS1=MSK6.A.SHIFT(NUPK(I),-30)$LG1=MSK8.A.SHIFT(NUPK(I),-36)
      DO 99 I2=1,2$IF(I2.EQ.2)LS1=LS2$IF(I2.EQ.2)LG1=LG2
      E23=E22=E21=15.
      IF(LS1.EQ.0.OR.LS1.EQ.22.OR.LS1.EQ.12.OR.LG1.GE.146)E22=E21=80.
      SIGMA1=SLCD(LS1+1,1)
      Z1=SLCD(LS1+1,2)*0.75
      IF(Z1.LT.2.)Z1=2.
      IF(Z1.LT.SLCD(LS1+1,3))Z1=SLCD(LS1+1,3)
      IF(SIGMA1.EQ..01)E22=E21=80.
```

```
          SIGMA2=0.5*SIGMA1$Z2=4.*Z1
           IF(E22.EQ.80.)SIGMA2=SIGMA1
          ILAT=2*I+I2-2
          SIGMA3=GLCD(LG1)$CALLGRIM(AMP,FAZ)$IAMP=AMP$IFAZ=FAZ
           NB=30*(2-I2)
       99 LEV=OR(SHIFT(IAMP,NB+11),SHIFT(IFAZ,NB),LEV)$NUPK(I)=LEV
      100 CONTINUE$END
```

SUBROUTINE GRIM

```
          SUBROUTINE GRIM(AMPIMP, FAZIMP)
    C-------F IS FREQUENCY IN HERTZ
    C-------PHI IS THE ANGLE OF INCIDENCE IN RADIANS
    C-------SIGMA1 IS THE CONDUCTIVITY OF THE UNSATURATED SOIL.
    C-------E21 IS THE DIELECTRIC CONSTANT OF UNSATURATED SOIL.
    C-------Z1 IS THE DEPTH IN METERS OF THE UNSATURATED SOIL)
    C-------SIGMA2 IS THE CONDUCTIVITY OF THE SATURATED SOIL.
    C-------E22 IS THE DIELECTRIC CONSTANT OF SATURATED SOIL.
    C-------Z2 IS THE DEPTH IN METERS OF THE SATURATED SOIL.
    C-------SIGMA3 IS THE CONDUCTIVITY OF BEDROCK.
    C-------E23 IS THE DIELECTRIC CONSTANT OF BEDROCK.
    C-------ZSOIL IS THE TOTAL DEPTH OF SOIL ABOVE BEDROCK (Z1 + Z2)
          COMPLEX T2N,T2D
          COMPLEX  A10, A20, A11, A12, A13, A21, A22, A23, A32, A33,
         1A34, A42, A43, A44, AK1, AK2, ETA1, ETA2, ETA12, ETA22,
         2ZETA1, ZETA12, ZETA2, ZETA22, TEMP, RENUM, REDEM, RE, PHI,BETA1
          COMPLEX  SAV1,SAV2,SAV3,SAV4,SAV5,SAV6
          COMPLEX  AK3,  ETA3,  ETA32,  ZETA3,  ZERO1, A54, A55, A64,
         1A65,A66, T2, T3N, T3D,T3    , A35, A45, A56
          COMMON  /THREE/ F    , PHI , SIGMA1, E21, AMU1, Z1, SIGMA2,
         1E22, AMU2, Z2, SIGMA3, E23, AMU3
          DATA  (C = 2.997925E8),            (TPI=6.2831853071796)
          DATA(  E0 = 8.854185336 7321E-12)
          DATA(PI = 3.141592654)  ,(PI2 = -1.570796327)
          DATA ISKIP/1/
          IF(ISKIP.EQ.0)GO TO 99
          ISKIP = 0
          ZERO1 = CMPLX (0.,1.)
        1 COSPHI = COS(PHI)
          SINPHI = SIN(PHI)
          COS2 = COSPHI * COSPHI
          SIN2 = SINPHI * SINPHI
          OMEGA = F * TPI
          ETA0=1.000338
          AK0=OMEGA/C*ETA0
          OMEGAE = OMEGA * E0
          EPSIL=1.E-15
       10 A10 = CMPLX(-COSPHI, 0.)
          A11 = A10
          A12 = A13 = A20 = A34 = A35 =        CMPLX(-1.,0.)
          A56 = CMPLX(1.,0.)
          A21 = CMPLX (1.,0.)
       99 AK2=AK0*CSQRT(CMPLX(E22,-SIGMA2/OMEGAE))
          ETA2 = AK2/AK0
```

```
      ETA22 = ETA2 * ETA2
      TEMP = ETA22 - SIN2
      ZETA2 = CSQRT (TEMP)
      A44 = ZETA2 *( SIN2 / (AMU2 * ETA22-SIN2) + 1.) /AMU2
      A45 = -A44
      AK3 = AK0 * CSQRT (CMPLX(E23,  -SIGMA3/OMEGAE))
      ETA3 = AK3 / AK0
      ETA32 = ETA3 * ETA3
      TEMP = ETA32 - SIN2
      ZETA3 = CSQRT (TEMP)
      A66 =  ZETA3 * (SIN2 / (AMU3 * ETA32 -SIN2) + 1.) / AMU3
   70 CONTINUE
      AK1 = AK0 *CSQRT(  CMPLX (E21, -SIGMA1/ OMEGAE))
   20 ETA1 = AK1/AK0
      ETA12 = ETA1 * ETA1
      TEMP = ETA12 - SIN2
      ZETA1 = CSQRT (TEMP)
      A22 = A23 = -ZETA1 *(SIN2/ (AMU1 * ETA12-SIN2 )  +1.) /AMU1
      A23 = -A23
      BETA1 = AK0 * ZETA1 * Z1*ZERO1
   30 A32 = CEXP(-BETA1)
      A33 = CEXP(BETA1)
      A42 =  A22 * A32
      A43 = A23 * A33
      BETA1 = AK0 * ZETA2 * Z2 * ZERO1
      A54 = CEXP (-BETA1)
      A55 = CEXP (BETA1)
      A64 = A44 * A54
      A65 = A45 * A55
      T2N=A54*A66-A64*A56
      T2D=A55*A66-A65*A56
      A1=A2=A3=A4=ABS(SIGMA2-SIGMA3)
      IF(((A1.LT.EPSIL ).AND.(A2.LT.EPSIL )).OR.
     1((A3.LT.EPSIL ).AND.(A4.LT.EPSIL )))GO TO 31
      T2=T2N/T2D
      GO TO 32
   31 CONTINUE
      T2=0.
   32 CONTINUE
      T3N = (A32 * A44 - A42 * A34) - (A32 * A45 - A42 * A35) * T2
      T3D = (A33 * A44 - A43 * A34) - (A33 * A45 - A43 * A35) * T2
      A2=A4=ABS(SIGMA2-SIGMA3)
      A1=A3=ABS(SIGMA1-SIGMA2)
      IF(((A1.LT.EPSIL ).AND.(A2.LT.EPSIL )).OR.
     1((A3.LT.EPSIL ).AND.(A4.LT.EPSIL )))GO TO 34
      IF(A1.LT.EPSIL)T3=T2*A32/A33
      IF(A1.GE.EPSIL)GO TO 36
      B1=REAL(T2)
      B2=AIMAG(T2)
      B3=REAL(T3)
      B4=AIMAG(T3)
  115 FORMAT(' ',8(E13.5,2X))
   36 CONTINUE
      IF(A1.LT.EPSIL)GO TO 35
      T3 = T3N / T3D
      GO TO 35
   34 CONTINUE
      T3=0.
   35 CONTINUE
      RENUM = (A10 * A22 - A20 * A12) - (A10 * A23 -A20 * A13) * T3
```

```
      REDEM = (A11 * A22 - A21 * A12) - (A11 * A23 - A21 * A13 ) * T3
      RE = RENUM / REDEM
      TEMP = (1. - RE)  /  (1.  +  RE)
      PMI = TEMP * COSPHI
   50 AMPIMP = CABS(PMI)
      FAZIMP = CANG (PMI)
      IF (FAZIMP.LE.PI2)   FAZIMP = FAZIMP + PI
      IF (FAZIMP. GE. -PI2)    FAZIMP = FAZIMP - PI
      IF(FAZIMP.LT.0.050)FAZIMP=0.050$IF(FAZIMP.GT.1.400)FAZIMP=1.400
      IF(AMPIMP.LT.0.001)AMPIMP=0.001$IF(AMPIMP.GT.0.200)AMPIMP=0.200
      AMPIMP=1000.*AMPIMP$FAZIMP=1000.*FAZIMP
      IF(FAZIMP.LT.0..OR.AMPIMP.LT.0.)PRINT 1776
 1776 FORMAT(*0  AMP OR PHASE NEGATIVE*)
      RETURN
      END
```

FUNCTION CANG

```
      FUNCTION CANG(Z)
C
C
      COMPLEX Z
C
C
C
C.....INITIALIZE CANG FOR QUADRANT CORRECTION, GET RE(Z) AND IM(Z), TEST
C.....X FOR + OR - TO FIND CORRECT HALF-PLANE...........................
      DATA  (C = 2.997925E8),            (TPI=6.2831853071796)
      PI2 = TPI
      PI = TPI /2.
      PIO2 = TPI / 4
      CANG=0.0
      X=REAL(Z)
      Y=AIMAG(Z)
      IF (X) 10,50,90
C.....X .LT. 0.0.........................................................
   10 IF (Y) 20,30,40
   20 CANG=-PI
      GO TO 90
   30 CANG=+PI
      RETURN
   40 CANG=+PI
      GO TO 90
C.....X .EQ. 0.0.........................................................
   50 IF (Y) 60,70,80
   60 CANG=-PIO2
      RETURN
C. . .X=0 AND Y=0 IS REALLY UNDEFINED, BUT IN AGREEMENT WITH CDC WE. . .
C. . .RETURN A VALUE OF CANG=0.0 . . . . . . . . . . . . . . . . . . . . .
   70 CANG=0.0
      RETURN
   80 CANG=+PIO2
C.....X .GT. 0.0 AND X .GT. 0.0 ADJUSTED FOR CORRECT QUADRANT...........
   90 CANG=ATAN (Y/X) + CANG
      RETURN
      END
```

SUBROUTINE SETUP

```
      SUBROUTINE SETUP
      COMMON/THREE/F,PHI,SIGMA1,E21,AMU1,Z1,SIGMA2,E22,AMU2,Z2,SIGMA3,
     .E23,AMU3
       COMMON/CODE/SLCD(65,3),G0,GLCD(148)
      DATA CFW/1.E-2/,CSW/5./,CWG/1.E-2/,CDG/1.E-4/
      TPI=8.*ATAN(1.0)$F=1.E5$PHI=80.*(TPI/360.)
      AMU1=AMU2=AMU3=1.
      DO 1 I=2,65
    1 READ 10,SLCD(I,1),SLCD(I,3),SLCD(I,2)
   10 FORMAT(10X,F10.1,F10.1,F10.1)
      DO2I=1,19$IB=(I-1)*8+1$IE=IB+7$IF(I.EQ.19) IE=148
    2 READ11,(GLCD(K),K=IB,IE)
   11 FORMAT(8F10.1)
      DO 3 I=2,65
      PRINT5,I,SLCD(I,1),SLCD(I,2)
    3 SLCD(I,1)=1./SLCD(I,1)
    5 FORMAT(I5,2F15.1)
      PRINT6$DO4I=1,148$PRINT5,I,GLCD(I)
    4 GLCD(I)=1./GLCD(I)
    6 FORMAT(*0   GEOLOGY RESISTIVITY*,//)
      GLCD(0)=SLCD(1,1)=CFW$SLCD(1,2)=20.
      SLCD(1,3)=20.
      SLCD(65,1)=CFW$SLCD(65,2)=20.
      SLCD(65,3)=20.
      END
```

5. DESCRIPTION OF PART I

5.1 Overview

The lithology and terrain of the LORAN coverage area are digitized from maps in PART I as illustrated in Figure 9. The required three maps are those which contain information on the type top soil or overburden, type or age of basement rock, and ground elevation above sea level. The first type of information is obtained from soil maps usually published by the country of the interested area. These are colored maps with a legend defining the various types of soil and associated depth. Geological maps are available for most areas of the world and the colored legend indicates the type of rock. Terrain maps with elevation data are readily available for the entire world. Fortunately, these have been previously digitized by the Defense Mapping Agency, and magnetic tapes with this data in proper sequence are available upon official request.

The following equipment was required for data digitization:

Calculator - HP9830A

Digitizer - HP9864A

Cassette Memory - HP9865A

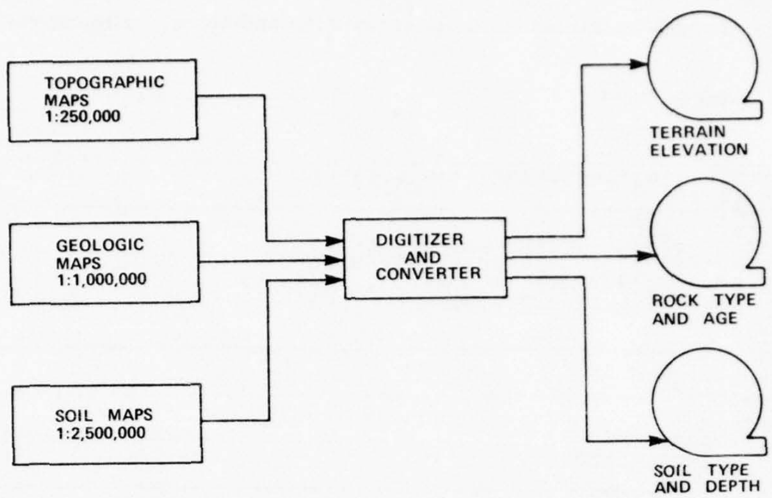


Figure 9. Formation of Data Base - Part I

The maps are digitized by recording, on a cassette tape, the contour table coordinates in inches using the southwest corner as the origin. The cursor cross hairs are moved around each constant color contour and points are generated in increments of one hundred to the inch. The table coordinates and geographic coordinates of the four corner points of the map are recorded for later use in conversion of the former to the latter. These cassette tapes are transcribed onto a disc file by way of the CDC 6600. The merged cassette data is then copied on to a 1/2-in. 2400-ft 800-bpi tape reel. The data on this tape is converted to LAT/LON and color number, and then into scan lines by PROGRAM GGCI. The scan line output format contains 24 bits for longitude, 24 bits for latitude, and 12 bits for color, making up the 60 bit CDC word. These points will then be sorted in PROGRAM SOR with the primary key being the latitude, secondary key the longitude, and tertiary key the color. PROGRAM DECODE sequences the sorted line points into a 30 arc second matrix for the final output tape of PART I. These programs are listed in Section B. Three such magnetic tapes are required, one for each of the ground properties. For a coverage area of 48 square degrees, there will be almost 700,000 data points recorded on each 7-track binary magnetic tape.

The Defense Mapping Agency generates similar data tapes in a program titled "Lineal Data to Terrain Matrix." This program is capable of forming a data matrix from map contour scans with a three arc second to a one minute spacing. By using the standard DMA format as input specification to PART II of the prediction program, DMA and locally generated tapes can be used interchangeably. The

present program is setup for a thirty arc second grid. Changes in this spacing will require changes in the incremental sequencing and storage allocations.

5.2 Program Listing for Part I

```
      PROGRAM SOR(TAPE3,TAPE2,TAPE4,TAPE1)
    9 READ(1) IP
      IF(EOF(1))2,1
    1 WRITE(2)IP$GO TO 9
    2 CONTINUE$REWIND 2$CALL SMSORT(10)
      CALLSMFILE("SORT","BINARY",2,"REWIND")
      CALL SMFILE("OUTPUT","BINARY",3,"REWIND")
      CALL SMKEY(1,1,10,0,"INTEGER")
      CALL SMEND
      REWIND 3
    5 READ(3)IP
      IF(EOF(3))4,3
    3 WRITE(4)IP
      GO TO 5
    4 STOP$END
MC6D,T150,CM77777,TP2.                                  2096     MCCOMISH
FTN,SL,A,P,BL,R=3,PL=20000.
VSN,TAPE1=OS0139.
REQUEST,TAPE1,HY,L,NR.
VSN,TAPE2=OS0172.
REQUEST,TAPE2,HY,L,RING.
FILE(TAPE2,RT=S,BT=C)
LDSET(FILES=TAPE2)
LGO.
      PROGRAM DECODE(INPUT,OUTPUT,TAPE6=OUTPUT,TAPE5=INPUT,TAPE1,TAPE2)
      COMMON IOUT(721),ICURPT,IMINH,IMAXH,IMINH2,IMAXH2,INFLOO
      DIMENSION Y(5001),IC(5001),IBUF(512)
      DIMENSION MONE(721),MTWO(721),MSAVE(721)

      ISTOP=0
       XXDEG=6.
      MCASE=1
      DO 11     I6=1,721
11    IOUT(I6)=0
      I20=513
      LEN=512
      ITOT=0

      XCUR=0.
      XCNT=XXDEG
      IMINH=0
      IMAXH=0
      IMINH2=-1000
      IMAXH2=-1000
      I99H1=-1000
      I99H2=-2000
      IE2=0
C SKIP TEST DATA
      IPHASE=1
15    CONTINUE
      IF(IE2.EQ.1) IPHASE=2
      J=1
      IE=0
      IFIRST=1

4     CONTINUE
      INFLOO = 0
C 4 IS TO READ IN ANOTHER SCAN LINE
      IF(IFIRST.EQ.1) GOTO6
      IF(XCUR.LT.(XXDEG-.004)) GO TO 6

C                  ** -DECODE MODIFICATIONS ... (START)
C                  ** -MODIFICATIONS (BRUCE HAINES 8/18/76)
```

```
      DO 8005 KA=1,721
 8005 MSAVE(KA)=IOUT(KA)
      IF(MCASE.GT.1) GO TO 8010
      DO 8007 KA=1,721
 8007 MONE(KA)=IOUT(KA)
      GO TO 8400
 8010 IF(MCASE.GT.2) GO TO 8030
      DO 8015 KA=1,721
 8015 MTWO(KA)=IOUT(KA)
      GO TO 8250
C                    ** -FOR NORMAL SCANS
 8030 DO 8100 KA=1,721
      IF(MTWO(KA).EQ.0) GO TO 8050
      IOUT(KA)=MTWO(KA)
      GO TO 8100
 8050 IF(IOUT(KA).EQ.0) GO TO 8070
      GO TO 8100
 8070 IF(MONE(KA).EQ.0) GO TO 8080
      IOUT(KA)=MONE(KA)
      GO TO 8100
 8080 IF(KA.EQ.1) GO TO 8100
      IOUT(KA)=IOUT(KA-1)
 8100 CONTINUE
      DO 8105 KA=1,721
      MONE(KA)=MTWO(KA)
      MTWO(KA)=MSAVE(KA)
 8105 CONTINUE
      GO TO 8300
C                    ** -FOR LAST SCAN LINE ONLY
 8170 DO 8200 KA=1,721
      IF(MTWO(KA).EQ.0) GO TO 8180
      IOUT(KA)=MTWO(KA)
      GO TO 8200
 8180 IF (MONE(KA).EQ.0) GO TO 8190
      IOUT(KA)=MONE(KA)
      GO TO 8200
 8190 IF(KA.EQ.1) GO TO 8200
      IOUT(KA)=IOUT(KA-1)
 8200 CONTINUE
      ISTOP=1
      GO TO 8300
C                    ** -
C                    ** -FOR FIRST SCAN LINE ONLY
 8250 DO 8270 KA=1,721
      IF(MONE(KA).EQ.0) GO TO 8255
      IOUT(KA)=MONE(KA)
      GO TO 8270
 8255 IF(MTWO(KA).EQ.0) GO TO 8260
      IOUT(KA)=MTWO(KA)
      GO TO 8270
 8260 IF(KA.EQ.1) GO TO 8270
      IOUT(KA)=IOUT(KA-1)
 8270 CONTINUE
C                    ** -
 8300 WRITE(6,612) XCURS
 612  FORMAT(1X,G21.14)
      WRITE(2) XCURS
      WRITE(6,610) IOUT
 610  FORMAT(25I4)
      WRITE(2) (IOUT(KLM),KLM=1,721)
 8400 XCURS=XCUR
      MCASE=MCASE+1
      IF(ISTOP.EQ.1) STOP
      IF(XCUR.GT.(XXDEG+.991)) GO TO 8170
C                    ** -DECODE MODIFICATIONS ... (FINISH)
```

```
6       CONTINUE
        XCNT=XCNT+(1./120.)
        IF (XCNT.LT.(XCUR+(1./240.))) GOTO 4
        IF(IPHASE.EQ.2) GOTO5
        IF(IE2.EQ.1) GOTO 15
5       IF (IE.EQ.1) GOTO 98
        DO 9 I5=1,5001
        Y(I5)=0.
        IC(I5) =0
9       CONTINUE
        IF(IFIRST.EQ.1) GOTO7
        XCUR=XNEW
        Y(1)=YSAVE
        IC(1)=ICSAVE
7       CONTINUE
   20   CONTINUE
        DO 10 I3=J, 5001
        IF(I3.LT.5001) GOTO2
        PRINT603,XCUR,Y(5000),IC(5000)
603     FORMAT(* ARRAYS Y AND IC TOO SMALL*,2G21.14,I5)
        STOP
2       CONTINUE
        IF(IPHASE.NE.1) GOTO 31
        READ(5,*)XNEW,Y(I3),IC(I3)
        IF(EOF(5)) 49,32

C READ IN A POINT.
31      CONTINUE
        READ(1) IP
        IF(EOF(1)) 50,1

1       XNEW=FLOAT(SHIFT(IP,-36))/120.
        IF(XNEW.LT.XCUR) GOTO31
        Y(I3)=FLOAT(SHIFT(IP,-12).AND. 7777 7777 B)*.001
        IC(I3)=IP.AND. 7777 B
        IF(IC(I3).GT.300) GOTO 220
C FOR GLITCHES
        IF((XNEW.LT.5).OR.(XNEW.GT.15)) GOTO 221
        IF((Y(I3).LT.47) .OR. (Y(I3).GT.55)) GOTO 221
32      IF(J.EQ.1.AND.I3.EQ.1) XCUR=XNEW
        IF(XCUR.NE.XNEW) GOTO 3
        IF(I3.EQ.1) GOTO 10
        IF (Y(I3) .LT. Y(I3-1)) GOTO 302
10      CONTINUE
        PRINT 604
604     FORMAT(* ERROR AT   604*)
        STOP
98      PRINT*," STMT NO. 98 ENCOUNTERED, FINISHED."
        STOP
220     PRINT*," IC.GT.300, PROBABLE GLITCHES."
        GOTO 31
221     WRITE(6,2601)
2601    FORMAT(" COORDINATES OUT OF RANGE. PROBABLE GLITCHES.")
        GOTO 31
302     WRITE(6,3601)
3601    FORMAT(6(" A POINT OUT OF ORDER"))
        Y(I3)=Y(I3-1)
        GOTO 10


C INITIALIZATION FOR PROCESSING
50      CONTINUE
        PRINT*,"END OF DATA.  ITOT=",ITOT
        IE=1
C IE=END OF FILE SWITCH
        GOTO1

49      IE2=1
3       CONTINUE
        IFIRST=0
        L=I3-1
C GO FROM 1 TO L WITH ARRAYS
        YSAVE=Y(I3)
        ICSAVE=IC(I3)
        Y(I3)=0.
        IC(I3)=0
        J=2
        IF(XCUR.LT.(XXDEG-.004)) GO TO 4
C PRINT SCAN LINE
```

```
C BEGIN PROCESSING(FLOWCHART)
C ICURPT IS  CURRENT POINT
      ISP1=0
      ISC1=0
      ISC2=0
      ICURPT=1
      ICURCL=IC(ICURPT)
25    CONTINUE
      IF(INFLOO.GT.400 ) GOTO 304
      GOTO 305
304   I3=I3+10
      ISP1=ISP1+10
      INFLOO=0
305   CONTINUE
      IF(ABS(Y(ICURPT+1)-Y(ICURPT)).LE..0151) GOTO 51
C IF WITHIN .015, GO TO 51
      GOTO 76
51    IF(IC(ICURPT+1).EQ.ICURCL) GOTO77
      ISP1=0
      GOTO 73
77    CONTINUE
      ICURPT=ICURPT+1
      GOTO 25
76    IF(ICURPT.GE.L) GOTO 4
73    ICURPT=ICURPT+1
      IF(IC(ICURPT).NE.ICURCL) GOTO 45
      CALL FILL(Y(ICURPT-1),Y(ICURPT),IC(ICURPT))
      IF(ABS(Y(ICURPT+1)-Y(ICURPT)).GT..0151) GOTO74
C IF NOT WITHIN .015, GO TO 74
      ICURPT=ICURPT+1
      IF(IC(ICURPT-1).EQ.IC(ICURPT)) GOTO 75
102   ICURCL=IC(ICURPT)
      GOTO 25
75    CONTINUE
      GOTO 102
C
45    CONTINUE
      ISC1=IC(ICURPT)
      IF(ABS(Y(ICURPT+1)-Y(ICURPT)).GT..0151) GOTO78
C IF NOT WITHIN .015, GO TO 78
      GOTO 79
78    IF(ICURPT.GE.L) GOTO 4
      ICURPT=ICURPT-1
      GOTO 74
79    CONTINUE
82    ICURPT=ICURPT+1
      IF(ICURCL.NE.IC(ICURPT)) GOTO 80
      GOTO81
80    CONTINUE
      IF((I99H1.EQ.I99H2).AND.(I99H2.EQ.ICURPT)) GOTO 105
      I99H1=I99H2
      I99H2=ICURPT
106   IF (ISP1.NE.0) GOTO 83
      ICURPT=ICURPT-1
      ICURCL=IC(ICURPT)
      GOTO25
105   ISP1=ISP1+1
      GOTO 106
83    ICURPT=ISP1
      ICURCL=ISC2
      GOTO25
81    CALL FILL(Y(ICURPT-2),Y(ICURPT),IC(ICURPT))
      ICURCL=ISC1
      GOTO25
C
C
72    CONTINUE
C HOLE SSSUMED(CASE1)
74    CONTINUE
      IF(ICURPT.GE.L) GOTO4
C IS THIS END OF SCAN LINE
C 4 IS TO READ IN ANOTHER SCAN LINE
      ICURPT=ICURPT+1
      ISP1=ICURPT-1
      CALL FILL(Y(ICURPT-1),Y(ICURPT),IC(ICURPT))
      IF(ABS(Y(ICURPT+1)-Y(ICURPT)).GT..0151) GOTO72
C IF NOT WITHIN .015, GO TO 72
      ICURPT=ICURPT+1
      ISC2=IC(ICURPT)
```

```
C (IN CASE BACKWARDS)
      ICURCL=IC(ICURPT)
      GOTO 25
      END
      SUBROUTINE FILL  (YMIN,YMAX,IC)
C FILLING IN COLOR NUMBERS
      COMMON IOUT(721),ICURPT,IMINH,IMAXH,IMINH2,IMAXH2,INFLOO
       INFLOO =INFLOO+1
      IF (YMIN.LT.48.) YMIN=48.
      IF(YMAX .GT. 54.) YMAX=54.
      IF(YMAX.LT.48.)  YMAX=48.
      IF(YMIN.GT.54.)YMIN=54.
      IF (YMIN.GT.YMAX) GOTO1
      IARYMIN=(YMIN-48.)*120.+1.+.5
      IARYMAX=(YMAX-48.)*120.+1.+.5
C IT IS ASSUMED THAT THE REAL VALUE WILL BE TRUNCATED
      IF((IARYMIN.EQ.IMINH).AND.(IARYMAX.EQ.IMAXH)
     2    .AND.(IMINH.EQ.IMINH2).AND.(IMAXH.EQ.IMAXH2)) GOTO 30
31    IMINH2=IMINH
      IMAXH2=IMAXH
      IMINH=IARYMIN
      IMAXH= IARYMAX
C FILL UP AN ARRAY OF LENGTH 721
      DO 1001 I=IARYMIN,IARYMAX
      IOUT(I)=IC
1001  CONTINUE
      RETURN
1     PRINT51
51    FORMAT(* ABORTED BECAUSE SUBROUTINE WAS CALLED WITH YMIN:YMAX*)
      STOP
30    ICURPT=ICURPT+3
      GO TO 31$END
MC64,T450,CM255000,TP2.                                 2096     MCCOMISH
FTN(SL,PL=99999)
VSN,TAPE1=GGC64C.
REQUEST,TAPE1,HY,L.
VSN,TAPE2=OS0139.
REQUEST,TAPE2,HY,L,RING.
LGO.
SKIP.
EXIT,S.
REWIND,OUTPUT.
REWIND,TAPE8.
COPYCF,TAPE8,OUTPUT.
      PROGRAM GGC1(INPUT,OUTPUT,TAPE8,TAPE1,TAPE2)
      COMMON/INPUT/IRECNO,NUMVEC, IXST,IYST,IVEC(6000),IRECTYP,IFEATNO
      COMMON IBLKNO,ISTVEC,IAGE,INOLD(1080),IRECRES,L,NOFTREC
      COMMON XP,YP,RCP(50),SLOMER(20),A(10),B(10),AA(10),BB(10),CC(10),
     Z DD(10),D4850,D5052,D5254
      DIMENSION NDXLAT(6000),NDYLONG(6000),LAT(6000),LONG(6000)
       DIMENSION NXX(5)
      DIMENSION R(6000),D1(6000),NGEO(6000),NOVEC(150),D2(6000)
      DIMENSION ICCNT(6000),SLOPE(6000)
    1 FORMAT (I4)
    2 FORMAT(*    NUMREC=*I4)
   94 FORMAT( 5(3X*LONGITUDE     LATITUDE*3X))
  106 FORMAT(*  TAPE HEADER*)
  112 FORMAT(*  RECORD NO.*I4,6X,*FEATURE NO.*I4,6X,*RECORD TYPE*I4,6X,*
     ABLOCK NO.*I4,4X,*RECORDING RESOLUTION=* I3)
  116 FORMAT(*  RECORD NO.*I4,6X,*FEATURE NO.*I4,6X,*RECORD TYPE*I4,6X,*
     1BLOCK NO.*I4)
  135 FORMAT(*   AGE=*I12,4X,I3* RECORDS IN THIS FEATURE*)
  145 FORMAT(* FIRST X=*I6,2X,              *FIRST Y=*I6,2X,
     A           *NUMVEC=*I5,2X,*STARTING VEC. POS. NO.=*I3)
  371 FORMAT(*  ERROR IN VECTOR* I5)
  920 FORMAT(9(1X,F12.8))
  921 FORMAT(7(2X,F11.3))
  930 FORMAT(7(2X,I6,1X,I6,1X,I3))
  934 FORMAT(*   ENDPOINTS OF VECTORS                Y=LONGITUDE, X= LAT.*)
  935 FORMAT(/* TOTAL NUMBER OF POINTS  =*I5)
  936 FORMAT(5(5X,*LONG*6X*LAT*7X))
  938 FORMAT(12(2X,I3,1X,I5))
  944 FORMAT(*  NUMBER OF POINTS IN FEATURE MAY EXCEED ARRAY SIZE*)
  950 FORMAT(4(1X,F8.0,1X,F8.6,1X,F7.0,1X,F7.0))
  951 FORMAT(10(1X,I5,1X,I6))
  952 FORMAT(6(2X,O20))
```

```
  994 FORMAT (* INCORRECT SLOPE COMPUTATION *)
      PHIR1=48.0   $    PHIR6=50.0   $    PHIR11=52.0   $    PHIR16=54.0
      THETA1=6.0 $ THETA2=8.0 $ THETA3=10. $ THETA4=12. $ THETA5=14.
      READ 1,NUMREC
      PRINT 2,NUMREC
      NOFILE=0
      IRECNO= -1
      DO 60 M=1,150
   60 NOVEC(M)=0
   70 DO 80 K=1,6000
      NDXLAT(K)=0
   80 NDYLONG(K)=0
C  READ IN NEW RECORD
  100 CALL INGGC6(IEOF,NOWORD)
      IF(IEOF.EQ.0)104,102
  102 NOFILE=NOFILE+1
      IF(NOFILE.GE.3)1000,100
  104 IF(IRECNO.GT.0)107,105
  105 PRINT 106
      GO TO 100
  107 IF(IRECNO.EQ.1)108,110
  108 SLOME 6 =SLOMER( 5)    $   A 6 =A( 5)    $     B 6=B(5)
      SLOME  7=SLOMER( 1)    $   A 7=A( 1)     $     B 7=B(1)
      SLOME  8=SLOMER( 6)    $   A 8=A( 6)     $     B 8=B(6)
      SLOME  9=SLOMER( 2)    $   A 9=A( 2)     $     B 9=B(2)
      SLOME  10=SLOMER( 7)   $   A10=A( 7)     $     B10=B(7)
      SLOME  11=SLOMER( 3)   $   A11=A( 3)     $     B11=B(3)
      SLOME  12=SLOMER( 8)   $   A12=A( 8)     $     B12=B(8)
      SLOME  13=SLOMER( 4)   $   A13=A( 4)     $     B13=B(4)
      SLOME  14=SLOMER( 9)   $   A14=A( 9)     $     B14=B(9)
      RCP48= (RCP(1)+RCP(2)+RCP(3)+RCP(4))/4.
      RCP50= (RCP(5)+RCP(6)+RCP(7)+RCP(8)+RCP(9))/5.
      RCP52= (RCP(10)+RCP(11)+RCP(12)+RCP(13)+RCP(14))/5.
      RCP54= (RCP(15)+RCP(16)+RCP(17)+RCP(18)+RCP(19))/5.
      PRINT 920,(SLOME6,SLOME7,SLOME8,SLOME9,SLOME10,SLOME11,SLOME12,
     Q           SLOME13,SLOME14)
      PRINT 921,(RCP48,RCP50,RCP52,RCP54,D4850,D5052,D5254)
  110 IF(IRECNO.EQ.2)111,114
  111 PRINT 112, IRECNO,IFEATNO,IRECTYP,IBLKNO,IRECRES
      GO TO 120
  114 PRINT 116, IRECNO, IFEATNO, IRECTYP, IBLKNO
  120 IF(IRECNO.GT.4)122,100
  122 IF(IRECTYP.EQ.30)130,140
  130 NT=0
      PRINT 135, IAGE,NOFTREC
      IF(NOFTREC.GT.6) PRINT 944
      GO TO 100
  140 IF(IBLKNO.EQ.1)141,142
  141 NDXLAT(1)= IXST     $      NDYLONG(1)= IYST
  142 PRINT 145, IXST,           IYST,           NUMVEC,ISTVEC
      N=NUMVEC
C
      DO 600 I=1,N
      J= I+NT
      K=ISTVEC + I
      IVEC(J)=INOLD(K)
      JVEC=IVEC(J) +1
      GOTO(210,220,230,240,250,260,270,280,290,300,310,320,330,340,350,3
     B60,370)JVEC
C   DELTA X IS THE INCREMENT OF LATITUDE FOR EACH SCALED VECTOR
C   DELTA Y IS THE INCREMENT OF LONGITUDE FOR EACH SCALED VECTOR
  210          IDELX=0 $ IDELY=2 $ GO TO 500
  220          IDELX=1 $ IDELY=2 $ GO TO 500
  230          IDELX=2 $ IDELY=2 $ GO TO 500
  240          IDELX=2 $ IDELY=1 $ GO TO 500
  250          IDELX=2 $ IDELY=0 $ GO TO 500
  260          IDELX=2 $ IDELY=-1$ GO TO 500
  270          IDELX=2 $ IDELY=-2$ GO TO 500
  280          IDELX=1 $ IDELY=-2$ GO TO 500
  290          IDELX=-1$ IDELY=2 $ GO TO 500
  300          IDELX=-2$ IDELY=2 $ GO TO 500
  310          IDELX=-2$ IDELY=1 $ GO TO 500
  320          IDELX=-2$ IDELY=0 $ GO TO 500
  330          IDELX=-2$ IDELY=-1$ GO TO 500
  340          IDELX=-2$ IDELY=-2$ GO TO 500
  350          IDELX=-1$ IDELY=-2$ GO TO 500
  360          IDELX=0 $ IDELY=-2$ GO TO 500
  370 PRINT 371,J
  372          IDELX=0 $ IDELY=0 $ GO TO 500
  500 NDXLAT(J+1)=NDXLAT(J) + IDELX
      NDYLONG(J+1)=NDYLONG(J) + IDELY
```

```
  600   CONTINUE
C
        NT=NT+NUMVEC
        IF(IBLKNO.LT.NOFTREC)100,602
   602  NOVEC(IAGE)=NOVEC(IAGE)+ NT
        NT2=NT +1
        PRINT 934
        PRINT 936
        PRINT 930,(NDYLONG(J),NDXLAT(J) ,IAGE,J=1,NT2)
        PRINT 935,NT2
C
        DO 701 J=1,NT2
C
C     ICONT IS THE PACKED WORD OF COORDINATES IN X-Y UNITS( LONG.=24 BITS, L
C     24 BITS, AGE=12 BITS, ALL RIGHT JUSTIFIED
   590  ICONT(J) = SHIFT(NDYLONG(J) , 36).OR.  SHIFT(NDXLAT(J),12).OR.IAGE
        R(J)=SCRT( (NDXLAT(J)-XP)**2 +(NDYLONG(J)-YP)**2 )
        IF(R(J).GT.RCP48)LAT(J)=(48.-2.0*(R(J)-RCP48)/(D4850        ))*1000.
        IF(R(J).GT.RCP50.AND.R(J).LE.RCP48)LAT(J)=(48.0+2.0*(RCP48-R(J))/
     A       (D4850          ) )*1000.
        IF(R(J).GT.RCP52.AND.R(J).LE.RCP50)LAT(J)=(50.0+2.0*(RCP50-R(J))/
     A       (D5052          ) )*1000.
        IF(R(J).GT.RCP54.AND.R(J).LE.RCP52)LAT(J)=(52.0+2.0*(RCP52-R(J))/
     A       (D5254          ) )*1000.
        IF(R(J).LE.RCP54)LAT(J)=(54.+2.0*(RCP54-R(J))/(D5254          ))*1000.
        SLOPE(J)=(YP-NDYLONG(J) )/ (XP-NDXLAT(J) )
        IF(SLOPE(J).GE.SLOME  6 .AND.SLOPE(J).LT.SLOME  7 ) 611,615
C     DIFFERENT FORM OF THE POINT-SLOPE DISTANCE COMPUTATION FORMULA--SAME RE
   611  D1(J)=ABS( (SLOME 6  *NDXLAT(J)       - NDYLONG(J)       +A 6 )/B 6 )
        D2(J)=ABS( (SLOME 7  *NDXLAT(J)       - NDYLONG(J)       +A 7 )/B 7 )
        LONG(J)= ( 6.0+D1(J)/(D1(J)+D2(J)))*1000
        GO TO 700
  615   IF(SLOPE(J).GE.SLOME  7 .AND.SLOPE(J).LT.SLOME  8 )  616,620
  616   D1(J)=ABS( (SLOME 7  *NDXLAT(J)       - NDYLONG(J)       +A 7 )/B 7 )
        D2(J)=ABS( (SLOME 8  *NDXLAT(J)       - NDYLONG(J)       +A 8 )/B 8 )
        LONG(J)= ( 7.0+D1(J)/(D1(J)+D2(J)))*1000
        GO TO 700
  620   IF(SLOPE(J).GE.SLOME  8 .AND.SLOPE(J).LT.SLOME 9  )  621,625
  621   D1(J)=ABS( (SLOME 8  *NDXLAT(J)       - NDYLONG(J)       +A 8 )/B 8 )
        D2(J)=ABS( (SLOME 9  *NDXLAT(J)       - NDYLONG(J)       +A 9 )/B 9 )
        LONG(J)= ( 8.0+D1(J)/(D1(J)+D2(J)))*1000
        GO TO 700
  625   IF(SLOPE(J).GE.SLOME  9 .AND.SLOPE(J).LT.SLOME 10 )  626,630
  626   D1(J)=ABS( (SLOME 9  *NDXLAT(J)       - NDYLONG(J)       +A 9 )/B 9 )
        D2(J)=ABS( (SLOME 10 *NDXLAT(J)       - NDYLONG(J)       +A 10)/B 10)
        LONG(J)= ( 9.0+D1(J)/(D1(J)+D2(J)))*1000
        GO TO 700
  630   IF(SLOPE(J).GE.SLOME 10 .AND.SLOPE(J).LT.SLOME 11 )  631,635
  631   D1(J)=ABS( (SLOME 10 *NDXLAT(J)       - NDYLONG(J)       +A 10)/B 10)
        D2(J)=ABS( (SLOME 11 *NDXLAT(J)       - NDYLONG(J)       +A 11)/B 11)
        LONG(J)= (10.0+D1(J)/(D1(J)+D2(J)))*1000
        GO TO 700
  635   IF(SLOPE(J).GE.SLOME 11 .AND.SLOPE(J).LT.SLOME 12 )  636,640
  636   D1(J)=ABS( (SLOME 11 *NDXLAT(J)       - NDYLONG(J)       +A 11)/B 11)
        D2(J)=ABS( (SLOME 12 *NDXLAT(J)       - NDYLONG(J)       +A 12)/B 12)
        LONG(J)= (11.0+D1(J)/(D1(J)+D2(J)))*1000
        GO TO 700
  640   IF(SLOPE(J).GE.SLOME 12 .AND.SLOPE(J).LT.SLOME 13 )  641,645
  641   D1(J)=ABS( (SLOME 12 *NDXLAT(J)       - NDYLONG(J)       +A 12)/B 12)
        D2(J)=ABS( (SLOME 13 *NDXLAT(J)       - NDYLONG(J)       +A 13)/B 13)
        LONG(J)= (12.0+D1(J)/(D1(J)+D2(J)))*1000
        GO TO 700
  645   IF(SLOPE(J).GE.SLOME 13 .AND.SLOPE(J).LT.SLOME 14 )  646,650
  646   D1(J)=ABS( (SLOME 13 *NDXLAT(J)       - NDYLONG(J)       +A 13)/B 13)
        D2(J)=ABS( (SLOME 14 *NDXLAT(J)       - NDYLONG(J)       +A 14)/B 14)
        LONG(J)= (13.0+D1(J)/(D1(J)+D2(J)))*1000
        GO TO 700
  650   IF(SLOPE(J).GE.SLOME14)651,655
  651   D1(J)=ABS( (SLOME 13 *NDXLAT(J)       - NDYLONG(J)       +A 13)/B 13)
        D2(J)=ABS( (SLOME 14 *NDXLAT(J)       - NDYLONG(J)       +A 14)/B 14)
        LONG(J)= (14.0+D2(J)/(D1(J)-D2(J)))*1000
        GO TO 700
  655   IF(SLOPE(J).LT.SLOME6)660,693
  660   D1(J)=ABS( (SLOME 6  *NDXLAT(J)       - NDYLONG(J)       +A 6 )/B 6 )
        D2(J)=ABS( (SLOME 7  *NDXLAT(J)       - NDYLONG(J)       +A 7 )/B 7 )
        LONG(J)= ( 6.0-D1(J)/(D2(J)-D1(J)))*1000
        GO TO 700
   693  PRINT 994
```

```
C
C     APPROXIMATION FOR LONG(J) WHOSE SLOPE IS UNDETERMINED
      LONG(J)=LONG(J-1)
C
 700  NGEO(J)=SHIFT(LONG(J),36).OR.SHIFT(LAT(J),12).OR.IAGE
 701  CONTINUE
C
C
C     PRINT 950,(R(J),SLOPE(J),D1(J),D2(J),J=1,NT2)
C     PRINT 94
C     PRINT 951,(LONG(J),LAT(J),J=1,NT2)
2237  FORMAT(*  *,5(I8,2X))
      WRITE(8,2237)IFEATNO,LONG(1),LAT(1),IAGE,NT2
      NXX(1)=IFEATNO$NXX(2)=LONG(1)$NXX(3)=LAT(1)$NXX(4)=IAGE$NXX(5)=NT2
      BUFFEROUT(2,1)(NXX(1),NXX(5))$IF(UNIT(2))1791,1791,1791
1791  BUFFEROUT(2,1)(NGEO(1),NGEO(NT2))$IF(UNIT(2))1792,1792,1792
1792  CONTINUE
      IF(IRECNO.LT.NUMREC)70,1000
1000  CONTINUE
      PRINT 938,(I,NOVEC(I),I=1,148)
      END
      SUBROUTINE INGGC6(IEOF,NOWORD)
C     THIS VERSION OF INGGC IS FOR THE 6 DEGREE TAPE
      COMMON/INPUT/IRECNO,NUMVEC, IXST,IYST,IVEC(6000),IRECTYP,IFEATNO
      COMMON IBLKNO,ISTVEC,IAGE,INOLD(1080),IRECRES,L,NOFTREC
      COMMON XP,YP,RCP(50),SLOMER(20),A(10),B(10),AA(10),BB(10),CC(10),
     Z DD(10),D4850,D5052,D5254
      DIMENSION JGGC(69),LX(20),LY(20),DL(19) ,DCP50(5),PHI(15)
      TYPE REAL LX,LY
    5 FORMAT(4(2X,P1E16.8))
    6 FORMAT(5(2X,P1E16.8))
    8 FORMAT(*  DISTANCES BETWEEN PARALLELS*/5(2X,E14.8) )
    9 FORMAT(4(2X,P1E10.4))
 143  FORMAT(* TABLE COORDINATES OF CONTROL POINTS,X(LATITUDE),Y(LONG)*)
 144  FORMAT(5(2X,I2,2(2X,F7.1)))
 146  FORMAT(5(2X,I2,2X,P1E18.10))
 200  FORMAT(2X,*XP=*P1E14.7,2X,*YP=*P1E14.7)
 436  FORMAT(*   FILE SUMMARY RECORD *)
 455  FORMAT(*    INVALID RECORD TYPE*)
 496  FORMAT(*   END OF FILE*)
1000  FORMAT(*  INVALID AGE NO.*)
1500  FORMAT(*  PARITY ERROR*)
C
C     READ NEW RECORD
 100  BUFFER IN (1,1) (JGGC(1), JGGC(69))
C     TEST IF EOF OR PARITY ERROR WAS ENCOUNTERED ON TAPE
 105  IF (UNIT(1)) 120,495,110
 110  PRINT 1500
 120  IEOF=0
      NOWORD = LENGTH(1)
      IRECNO = IRECNO+ 1
      IF(IRECNO.GT.0)130,500
 130  CALL UNPACK4(JGGC,69)
      IFEATNO = INOLD(7)*16      + INOLD(8)+ INOLD(6)*256
      IRECTYP = INOLD(9)*(2**4) + INOLD(10)
      IBLKNO= INOLD(11)*(2**4) + INOLD(12)
      IF(IRECNO.EQ.1)135,150
 135  DO 140 I=1,19
      J=8*I +253
      LX(I)= INOLD(J)*4096+INOLD(J+1)*256+INOLD(J+2)*16+INOLD(J+3)
 140  LY(I)=INOLD(J+4)*4096 +INOLD(J+5)*256 +INOLD(J+6)*16 +INOLD(J+7)
      PRINT 143
      PRINT 144,(I,LX(I),LY(I),I=1,19)
      XPN=(LX(5)*LX(19))*(LY(15)-LY(9))+(LX(5)*LX(9))*(LY(19)-LY(15))+
     A   (LX(9)*LX(15))*(LY(5)-LY(19))+(LX(15)*LX(19))*(LY(9)-LY(5))
      XPD=(LY(15)-LY(5))*(LX(19)-LX(9))-(LX(15)-LX(5))*(LY(19)-LY(9))
      XP=XPN/XPD
      YP =LY(5)+(XP -LX(5))*(LY(15)-LY(5))/(LX(15)-LX(5))
      DO 1441 M=5,14
1441  DL(M)=SQRT((LX(M+5)-LX(M))**2 +  (LY(M+5)-LY(M))**2 )
      D4850=         D5052=(DL(5)+DL(6)+DL(7)+DL(8)+DL(9))/5.
      D5254=(DL(10)+DL(11)+DL(12)+DL(13)+DL(14))/5.
      PRINT 8,(DL(M),M=5,14)
      PRINT200, XP,YP
      DO 145 I=1,19
 145  RCP(I)=SQRT(  (LX(I)-XP)**2 +  (LY(I)-YP)**2 )
      PRINT 146,(I,RCP(I),I=1,19)
      DO 1443 M=1,4
      DCP50 (M)=SQRT((LX(M+5)-LX(M+4))**2 +  (LY(M+5)-LY(M+4))**2 )
```

```
 1443 PHI(M)=( ASIN(DCP50(M)/RCP(M+4)))*(180./3.14159)
      PRINT9,(PHI(M),M=1,4)
      DO 147 M=1,9
      AA(M)=YP-LY(M)   $ BB(M)=LX(M)-XP   $ CC(M)=XP*LY(M)-LX(M)*YP
      DD(M)=SQRT(AA(M)**2 +BB(M)**2)
      A(M) =((LX(M  )*YP)-(LY(M  )*XP))/(LX(M  )-XP)
  147 B(M)=SQRT((((LY(M   )-YP)/(LX(M   )-XP))**2)+1.)
      DO 148 M=1,19
  148 SLOMER(M)=(LY(M)-YP)/(LX(M)-XP)
      PRINT 5,(SLOMER(M),M=1,4)
      PRINT 6,(SLOMER(M),M=5,19)
 150  IF(IRECNO.EQ.2)IRECRES=INOLD(23)*(2**4)  + INOLD(24)
      IF(IRECNO.GT.4)160,500
  160 IF(IRECTYP.EQ.30) 162,189
 162  JH = INOLD(45)*(2**4)  + INOLD(46)
      JT = INOLD(47)*(2**4)  + INOLD(48)
      JU=INOLD(49)*(2**4)  + INOLD(50)
      NOFTREC=IBLKNO  - 1
      IF(JU.GT.175.AND.JU.LT.186)170,176
  170 IF(JT.GT.175.AND.JT.LT.186)172,188
 172  IF(JH.GT.176.AND.JH.LT.186)174,188
 174  IAGE=(JH-176)*100 + (JT-176)*10 + (JU-176)
      IF(IAGE.GT.0.AND.IAGE.LT.149)500,188
 176  IF(JU.EQ. 32.OR.JU.EQ.160)177,188
 177  IF(JT.GT.175.AND.JT.LT.186)178,180
 178  IF(JH.GT.176.AND.JH.LT.186)179,188
 179  IAGE=(JH-176)*10 +(JT-176)
      IF(IAGE.GT.0.AND.IAGE.LT.100)500,188
 180  IF(JT.EQ.32.OR.JT.EQ.160)181,188
 181  IF(JH.GT.176.AND.JH.LT.186)182,188
 182  IAGE= JH-176
      IF(IAGE.GT.0.AND.IAGE.LT.10 )500,188
 188   IAGE=0
      PRINT 1000
      GO TO 500
 189  IF(IRECTYP.EQ.31) 190,430
C     STARTING VECTOR POSITION NUMBER USUALLY "28"
 190  ISTVEC=(INOLD(14).AND.07B)*(2**8) +INOLD(15)*(2**4) +INOLD(16)
      NUMVEC=(INOLD(18).AND.07B)*(2**8) +INOLD(19)*(2**4) + INOLD(20)
      IXST =(INOLD(21)*(2**12) + INOLD(22)*(2**8) + INOLD(23)*(2**4)
     1  + INOLD (24))
      IYST =(INOLD(25)*(2**12) + INOLD(26)*(2**8) + INOLD(27)*(2**4)
     1 + INOLD(28))
      L=NOFTREC-IBLKNO
      GO TO 500
 430  IF(IRECTYP.EQ.90)435,450
 435  PRINT 436
      NUMFEAT=INOLD(21)*(2**12)+INOLD(22)*(2**8)+INOLD(23)*(2**4)+INOLD(
     $24)
      NUMBLKS=INOLD(25)*(2**12)+INOLD(26)*(2**8)+INOLD(27)*(2**4)+INOLD(
     $28)
      GO TO 500
  450 PRINT 455
      GO TO 500
  495 IEOF=1
      PRINT 496
  500 RETURN
      END
      SUBROUTINE UNPACK4(JGGC,NWD)$DIMENSION JGGC(69)
      COMMON IBLKNO,ISTVEC,IAGE,INOLD(1080),IRECRES,L,NOFTREC
      COMMON XP,YP,RCP(50),SLOMER(20),A(30),B(30),AA(10),BB(10), C(30),
     Z  D(30),D4850,D5052,D5254,DEGLONG(50),DEGLAT(50)
      INX=0$DO 2 J=1,NWD$DO 2 I=1,15$INX=INX+1
    2 INOLD(INX)=17B.A.SHIFT(JGGC(J),-4*(15-I))
   10 FORMAT(5022)
    9 FORMAT(*0  INOLD*)
      END
6999
MC6GS,T160,CM145000,TP2.                               2096     MCCOMISH
FTN,SL,A,P,BL,R=3,PL=49999,T.
MAP(ON)
VSN,TAPE1=OS0139.
REQUEST,TAPE1,HY,L.
VSN,TAPE4=OS0172.
```

6. DESCRIPTION OF PART IV

6.1 Overview

The purpose of PART IV of the LORAN Grid Prediction Program is to update the input data base to PART II. By comparing the time differences at measured and calculated fixed points in the service area, corrections are applied to the postulated top soil conductivity values to zero out the difference between the former. A technique described by Elkins[22] has been utilized which employs a continuous Kalman filter with updating of references state vector. The details of this program are described in Program ESTIMAE.[23] It is pointed out that of the nine parameters required to calculate the surface impedance, only the two predominant top soil conductivities are altered.

The approach taken for system tuning or data base updating is as follows:

(1) Compute time difference in PART III of Program for locations for which measured data is available.

(2) Identify two values for top soil conductivity that occur most frequently in the geographical area of interest.

(3) The variation in the LORAN time difference due to a variation in topsoil conductivity is computed by changing in turn, each of the two conductivity values where they occur in the geographical area of interest and proceeding with the step outlined in "1" above.

(4) The difference between the observed and calculated data, the rates of change in time difference due to changes in each conductivity value, estimate of the error in the initially chosen values of conductivity, and estimates of the error in the observations are input to a Kalman filter program to produce an improved estimate of the value of conductivity.

(5) The improved values of conductivity are written into the data base.

For the test program, the conductivity was changed by 10 percent to form the required partial derivatives; absolute conductivity was estimated within a factor of two, and the standard deviation of the time difference measurement was assumed to be 100 μsec. Utilizing forty test points, this technique reduced the prediction errors by 33 percent.

A listing of the program is presented below.

---

22. Elkins, T.E. (1976) Empirical Correction of Soil Conductivity Model, RADC/ET Private Communication.

23. Program ESTIMAE - Problem No. 4723 (1975) Analyses and Simulation Branch, AFCRL dated 1 November 1975.

```
1 ESTMATE      74/74    OPT=1                                    FTN 4.5+414  06/26/77
      PROGRAM ESTMATE(INPUT=401B,OUTPUT=401B,TAPE1=401B,TAPE2=401B,
     1 TAPE3=401B,TAPE4=401B,TAPE10,TAPE11,TAPE20=1101B)
C.... GIVEN (1) A SET OF DIFFERENTIAL EQUATIONS DESCRIBING A DYNAMICAL
C  SYSTEM, (2) AN ESTIMATE OF THE STATE OF THE SYSTEM AT SOME INITIAL
C  TIME ALONG WITH STATISTICS PROVIDING A MEASURE OF THIS ESTIMATE,
C  AND (3) A DATA FILE CONTAINING OBSERVATIONS BEARING ON THE STATE OF
C  THE SYSTEM AT SPECIFIED POINTS OF TIME.  COMPUTE THE BEST ESTIMATE OF
C  THE STATE OF THE SYSTEM ALONG WITH SOME STATISTICS GIVING A MEASURE
C  OF THE ESTIMATE AT EACH OF THE SPECIFIED POINTS OF TIME.  THE BEST
C  ESTIMATE IS BASED ON ALL OBSERVATIONS PROCESSED UP TO THE CURRENT
C  TIME. A BACKWARD SMOOTHING FEATURE IS PROVIDED AS AN OPTION SO THAT
C  THE ESTIMATES AT THE FIRST K POINTS OF TIME WILL REFLECT THE
C  INCLUSION OF ALL OBSERVATIONS UP TO THE K TH POINT OF TIME.
      COMMON/CASE/CASE
      COMMON/RK/TI,TF,NRK4578,SDT,DT,TOL,SP,NERR1,NERR2,ORD,
     1 NSTEP,NREJ,IS121
      COMMON/STREF/IXR,XR(1)
      COMMON/STESTM/IXE,XE(1)
      COMMON/STRSDL/IX,X(1)
      COMMON/STCOV/IP,JP,P(1)
      COMMON/STNCOV/ IQ,QCOV
      COMMON/OBSH/IH,JH,H(1)
      COMMON/OBSY/IY,Y(1)
      COMMON/OBSCOV/IR,R(1)
      COMMON/MEASCOV/RCOV(6)
      COMMON/PARAMS/KSTRSDL,HSTEP,TIMETOL,KPUNCH,KREWIND,TOLRNCE,IDER,
     1 KPLOT,KEND,KONE,KS,KSMOOTH,MAXS,MAXKC,MPTY,NGPS,NLRSR,NLRSW,
     2 NFILF,NFILB,NSAT,NNSAT,NGPS1,NGPS2,NRSEQ,NRRAN,NWSEQ,NWRAN,NPRAN,
     3 IOB,JZH,IST,IS1,IS2,KCRIT,KCBEGIN,KPRINT,KWRITE,KDATA,KTYPES,
     4 MAXDATA,IA,JA,KA,IM,JM,KM,MD,KPLWD,ITYPES(2),PRINMOM(3),KQKLOCK,
     5 IDERSET
C.... THE FOLLOWING COMMON AND EQUIVALENCE STATEMENTS ARE DEPENDENT ON
C  THE DATA FILE USED.
      COMMON/BUFF/LBUFF(4),TIME
      DIMENSION ID(1)  $  EQUIVALENCE (D,ID)
      COMMON D(36),XR0(12),S(144)
      DIMENSION X0(12),DIAG0(12)
      EXTERNAL DER
      NAMELIST/VALUES/CASE,TSTART,TSTOP,X0,DIAG0,KPLOT,KPRINT,
     1 KWRITE,KREWIND,KSMOOTH,MAXS,KCRIT,MAXKC,KCBEGIN,KTYPES,MAXDATA,
     2 IA,JA,KA,IM,JM,KM,NRK4578,STEP,TOLRNCE,SP,NERR1,NERR2,ORD,IST,
     3 IS1,IS2,JZH,KSTRSDL,QCOV,RCOV,TIMETOL,KPARTS,KPLWD,MD,RUNMAX,
     4 KQKLOOK
C.... READ IN INITIAL PARAMETERS.  CALL SUBROUTINES TO SET INITIAL
C  PARAMETERS AND TO SELECT OPTIONS.
    1 CASE=CASE+1.  $  READ VALUES
      IF(CASE.LT.0.)46,2
    2 CALL FILEIO(T)
      CALL OBSERVE
      CALL STATS
      CALL PACK(KPLWD,I,I)
      SDT=STEP  $  TOL=TOLRNCE
      IDER=0
C.... SET REFERENCE STATE VECTOR AND DIAGONAL STATE COVARIANCE MATRIX TO
C  INITIAL CONDITIONS.
      CALL DIAGNAL(IST,X0,XR,DIAG0,IP,P)
C.... SET MAXIMUM RUN-TIME LIMIT.
```

```
      CLOCK0=SECOND(T)
C.... PRINT INITIAL PARAMETERS.
      PRINT VALUES
C.... SET START AND STOP TIMES.
      TI=TSTART
      IST1=IST+1  $  IST21=IST*IST1
C.... READ IN THE NEXT DATA RECORD.
    3 CALL FILEIO1(TI), RETURNS(4,42)
    4 IF(TIME.GT.TSTOP)5,8
    5 KEND=1  $  GO TO 42
C.... TIME OF DATA RECORD LIES WITHIN REQUESTED INTERVAL.
    8 T=TIME  $  KOP=0
   11 TF=T
      HSTEP=ABS(TF-TI)
C.... TEST WHETHER OR NOT TI AND TF ARE APPROXIMATELY EQUAL.
   14 IF(HSTEP.LE.TIMETOL)21,15
C.... INTEGRATE FORWARD TO TF.
   15 IF(KOP)19,16
C.... MAP THE STATE COVARIANCE MATRIX FORWARD.
   16 CALL MATRIX(0,IST,IST,0,P,IP,XR(IST1),IST,0,0)
      IDERSET=2  $  CALL RK4578(XR,DER)
      CALL MATRIX(0,IST,IST,0,XR(IST1),IST,P,IP,0,0)
      GO TO 21
   19 IF(KRESET.EQ.1) GO TO 20  $  KRESET=1
      CALL MOVLEV(XR,XR0,IST)
   20 CALL RK4578(XR,DER)
   21 TI=TF
C.... SELECT THE DATA FROM THE RECORD.
      IF(KOP.EQ.0)22,25
   22 CALL OBSERV1  $  IOB=0  $  ITYPES(1)=ITYPES(2)=1H
   23 KOP=KOP+1  $  IF(KOP.GT.KDATA)36,24
   24 N=MD*(KOP-1)+1  $  T=D(N)  $  I=ID(N+2)
      IF(KOP.EQ.1)80,11
C.... INITIALIZE STATE TRANSITION MATRIX.
   80 DO 10 M=1,IST  $  L=IST*M  $  DO 9 K=1,IST  $  J=K+L
    9 XR(J)=0.  $  J=M+L
   10 XR(J)=1.
      KRESET=0
      IDERSET=1  $  GO TO 11
C.... ENTER APPROPRIATE SUBROUTINE TO FIND THE COMPUTED OBSERVATION, THE
C  DIFFERENCE BETWEEN THE GIVEN OBSERVATION AND THE COMPUTED
C  OBSERVATION, AND THE PARTIALS OF THE COMPUTED OBSERVATIONS.
   25 IF(I.EQ.1)26,27
   26 CALL SSENSOR(KPARTS,D(N),S)  $  GO TO 35
   27 IF(I.EQ.2)28,29
   28 CALL ESENSOR(KPARTS,D(N),S)  $  GO TO 35
   29 IF(4.LE.I.AND.I.LE.6)30,31
   30 CALL MAGDATA(KPARTS,D(N),S)  $  GO TO 35
   31 IF(I.EQ.16)34,36
   34 CALL ACCEL(KPARTS,D(N),S)
C.... SAVE OBSERVATION TYPES FOR PRINTOUT.
   35 IF(KPRINT.AND.512)530,23
  530 J=3*KOP-1  $  ENCODE(16,531,S) J
  531 FORMAT(10H(T1,2A10,T,I2,4H,I2))
      ENCODE(20,S,ITYPES) ITYPES,I
      GO TO 23
   36 TI=TIME  $  IF(IOB)37,41
```

```
   37 IF(KRESET.EQ.1) CALL MOVLEV(XRO,XR,IST)
C.... ENTER THE FILTER.
      IF(KONE)38,39,39
   38 NFILB=NFILB+1  $  GO TO 40
   39 NFILF=NFILF+1
   40 CALL FILTER, RETURNS(44)
C.... COMPUTE THE BEST ESTIMATE.
      CALL MATRIX(21,IST,1,0,XR,IXR,X,IX,XE,IXE)
C.... DETERMINE PRINCIPAL VALUES OF ANGLES.
 4000 DO 400 I=1,3
  400 XE(I)=PRINVAL(XE(I))
C.... REINITIALIZE STATE REFERENCE VECTOR.
      CALL MOVLEV(XE,XR,IST)
C.... CHECK FOR MAXIMUM RUN TIME.
      CLOCK=SECOND(T)-CLOCK0  $  IF(CLOCK.GT.RUNMAX) KEND=1
C.... GATHER APPROPRIATE STATISTICS ON THE ESTIMATION.
      CALL STATS1
      IF(KEND.EQ.0) CALL FILEIO2(TI), RETURNS(4,42)
C.... INITIATE END PROCESSING.
   42 CALL FILEIO3(T)
   43 GO TO 1
   44 PRINT 45,NGPS,KONE
   45 FORMAT(*0SINGULAR MATRIX ENCOUNTERED.*5X*GROUP=*I4,5X*KONE=*I2)
C.... SIMPLY EQUATE BEST ESTIMATE VECTOR AND STATE REFERENCE VECTOR.
   41 CALL MOVLEV(XR,XE,IST)  $  DO 450 K=1,IST
  450 X(K)=0.  $  GO TO 4000
   46 CONTINUE  $  END
```



```
      FUNCTION PRINVAL(ANGLE)
C.... PRINVAL RETURNS THE PRINCIPAL VALUE OF THE GIVEN ANGLE.
C     ( - PI .LT. PRINVAL .LE. PI )
      DATA PI/3.14159265358979З/
      A=ANGLE
    1 AA=ABS(A)  $  IF(AA.LE.PI) GO TO 2
      A=A-SIGN(2.*PI,A)  $  GO TO 1
    2 PRINVAL=A  $  RETURN  $  END
```

```
      SUBROUTINE STATS
C.... RECORDS STATISTICS IN ORDER TO EVALUATE THE PERFORMANCE OF THE
C  ESTIMATION PROCESS.  CONTROLS THE INITIATION OF THE BACKWARD
C  SMOOTHING PROCESS AND THE ASSOCIATED RANDOM FILE WRITING.  CONTROLS
C  ALL PRINTED OUTPUT ON FILES 1,2,3, AND 4.
      COMMON/STREF/IXR,XR(1)
      COMMON/STESTM/IXE,XE(1)
      COMMON/STRSDL/IX,X(1)
      COMMON/STCOV/IP,JP,P(1)
      COMMON/STCOVD/IPD,D(1)
      COMMON/OBSY/IY,Y(1)
      COMMON/PARAMS/IQ(7),KPLOT,KEND,KONE,KS,KSMOOTH,MAXS,MAXKC,MPTY,
     1 NGPS,NLRSR,NLRSW,NFILF,NFILB,NSAT,NNSAT,NGPS1,NGPS2,NRSEQ,NRRAN,
     2 NWSEQ,NWRAN,NPRAN,IOB,JZH,IST,IS1,IS2,KCRIT,KCBEGIN,KPRINT,KWRITE
     3 ,JQ(11),OBTYPES(2)
      COMMON/CRIT/CRIT(1)
C.... THE FOLLOWING COMMON AND EQUIVALENCE STATEMENTS ARE DEPENDENT ON
C  THE DATA FILE USED.
      COMMON/BUFF/LBUFF(4),W(127)
      DIMENSION XT(1)  $  EQUIVALENCE (W,TIME),(W(10),XT)
      COMMON DX(12),XD(12),IS(24),D(50)
      DIMENSION S(1)  $  EQUIVALENCE (IS,S)
      LOGICAL NRSEQ,NRRAN,NWSEQ,NWRAN,NPRAN
      DATA MAXK/5/,MORE/5H MORE/
C.... SET INITIAL CONSTANTS.
      NLRSR=NLRSW=NGPS=NFILF=NFILB=NSAT=NNSAT=KST=KEND=0
      NRSEQ=.T.  $  NRRAN=NWSEQ=NWRAN=NPRAN=.F.
      KS=KSMOOTH  $  KONE=1
C.... SET PRINT OPTIONS.  AT MOST 1 OPTION PER FILE IS ALLOWED.
C  KPRINT BIT 0.  PRINT TIME, STATE VECTOR ON FILE 1.
C  KPRINT BIT 1.  PRINT TIME, STATE RESIDUALS VECTOR ON FILE 2.
C  KPRINT BIT 2.  PRINT TIME, STATE DEVIATIONS VECTOR ON FILE 2.
C  KPRINT BIT 3.  PRINT TIME, STATE DERIVATIVES VECTOR ON FILE 2.
C  KPRINT BIT 4.  PRINT TIME, DIAGONALS OF STATE COVARIANCE MATRIX ON
C  FILE 3.
C  KPRINT BIT 5.  PRINT TIME, STATE COVARIANCE MATRIX ON FILE 3.
C  KPRINT BIT 6.  PRINT TIME, TORQUES ON FILE 3.
C  KPRINT BIT 7.  PRINT COMPLETE STATISTICS ON FILE 4.
C  KPRINT BIT 8.  PRINT PARTIAL STATISTICS ON FILE 4.  (KCRIT.NE.0)
C  KPRINT BIT 9.  PRINT TIME, SOME STATISTICS, AND OBSERVATION RESIDUALS
C  VECTOR ON FILE 4.
      IF(KPRINT)2,1
    1 ASSIGN 55 TO M1  $  GO TO 17
    2 CALL WRITERD(4,0,0)
      ASSIGN 41 TO M1  $  ASSIGN 46 TO M2
      ASSIGN 52 TO M3  $  ASSIGN 55 TO M4
      ASSIGN 541 TO M41
C.... SET PRINT LIMITS ON STATE VECTOR ELEMENTS.
      IS2=MIN0(12,IS2,IST,(IX-IS1+1))  $  IS4=IS1+IS2-1  $  IS3=IS2+1
      PRINT 16,(I,I=IS1,IS4)
      PRINT 160  $  PRINT 161
      I=J=1
    3 K=KPRINT.AND.J  $  IF(K.NE.0) GO TO (5,5,7,9,10,11,13,14,14,15),I
    4 J=2*J  $  I=I+1  $  IF(I.GT.10)17,3
    5 ASSIGN 40 TO M1  $  GO TO 4
    6 ASSIGN 42 TO M2  $  GO TO 4
    7 ASSIGN 43 TO M2  $  GO TO 4
```

```
    9 ASSIGN 44 TO M2  $  GO TO 4
   10 ASSIGN 47 TO M3  $  GO TO 4
   11 ASSIGN 48 TO M3  $  GO TO 4
   13 ASSIGN 51 TO M3  $  GO TO 4
   14 KST=1
  140 ASSIGN 54 TO M4
      IF(I.EQ.9) ASSIGN 53 TO M4  $  GO TO 4
   15 ASSIGN 540 TO M41  $  GO TO 140
   16 FORMAT(*1DESCRIPTION OF PRINTOUTS ON FILES 1, 2, OR 3*/5X*ONLY THE
     1 FOLLOWING COMPONENTS OF THE STATE VECTOR, STATE RESIDUALS VECTOR,
     2 STATE DEVIATIONS VECTOR, STATE DERIVATIVES VECTOR OR THE*/* FOLLO
     3WING COLUMNS OF THE STATE COVARIANCE MATRIX OR THE FOLLOWING DIAGO
     4NAL ELEMENTS OF THE STATE COVARIANCE MATRIX ARE PRINTED OUT.*/1X,
     512I5)
  160 FORMAT(* FOR THE OBSERVATION RESIDUALS VECTOR, ALL COMPONENTS, AS
     1WELL AS THE OBSERVATION TYPES, ARE PRINTED OUT.*)
  161 FORMAT(
     6*-DESCRIPTION OF STATISTICS PRINTOUT ON FILE 4*/5X*GP....POSITION
     7OF DATA GROUP WRT FIRST DATA GROUP PROCESSED*/5X*LR R....LOCATION
     8OF LOGICAL RECORD OF DATA WRT FIRST LOGICAL RECORD READ*/5X
     9*LR W....CUMULATIVE NUMBER OF LOGICAL RECORDS WRITTEN TO OUTPUT DA
     ATA FILE          */5X*F F....CUMULATIVE NUMBER OF TIMES ENTERED FIL
     BTER FOR FORWARD FILTERING*/5X*F B....CUMULATIVE NUMBER OF TIMES EN
     CTERED FILTER FOR BACKWARD SMOOTHING*/5X*SAT....CUMULATIVE NUMBER O
     DF TIMES PERFORMANCE CRITERIA SATISFIED*/5X*NSAT....CUMULATIVE NUMB
     EER OF TIMES PERFORMANCE CRITERIA NOT SATISFIED*/5X*ELEMENT....RELA
     FTIVE LOCATION WITHIN STATE OR STATE RESIDUAL VECTOR*/5X*VALUE....T
     GEST VALUE WHICH EXCEEDED SPECIFIED CRITERIA*/5X*MORE....FLAG IMPLY
     HING THAT MORE ELEMENTS FAILED TEST*/5X*TIME....TIME OF DATA GROUP*
     I)
   17 ASSIGN 18 TO M6
C.... SET CRITERIA OPTIONS.
C  KCRIT BIT 0.  COVARIANCE LIMIT CRITERIA.
C  KCRIT BIT 1.  COMPARISON WITH EXISTING STATE CRITERIA.
      ASSIGN 37 TO M5
      IF(KCRIT.EQ.1) ASSIGN 21 TO M5
      IF(KCRIT.EQ.2) ASSIGN 22 TO M5
C....SET SEQUENTIAL WRITE OPTION.
      IF(KWRITE.EQ.1) NWSEQ=.NOT.NWSEQ
      RETURN
      ENTRY STATS1
C.... USUAL ENTRY POINT INTO STATS.
      GO TO M6,(18,20)
C.... WHEN PROCESSING FIRST DATA GROUP, SPECIFY NECESSARY OPTIONS.
   18 ASSIGN 20 TO M6
      IF (KPLOT.EQ.1) NPRAN=.NOT.NPRAN
      IF(KSMOOTH)19,20
   19 NWRAN=.NOT.NWRAN  $  NWSEQ=.NOT.NWSEQ  $  NPRAN=.NOT.NPRAN
      LGPNS=0
C.... CHECK WHETHER OR NOT CRITERIA ARE SATISFIED.
   20 K=0  $  DO 25 I=1,IST  $  GO TO M5,(21,22,37)
   21 J=1+IP*(I-1)  $  VAL=P(J)  $  GO TO 220
   22 VAL=XE(I)-XT(I)
  220 IF(ABS(VAL).LE.CRIT(I))25,23
   23 K=K+1  $  IF(KST)24,36
C... SAVE DATA ON ELEMENTS NOT SATISFYING CRITERIA.
   24 IS(K)=I  $  K=K+1  $  S(K)=VAL  $  IF(K.GT.2*MAXK)25,26
```

```
   25 K=K-1  $  IS(K)=MORE  $  GO TO 36
   26 CONTINUE  $  IF(K)36,27
   27 NSAT=NSAT+1  $  IF(KS.EQ.1)28,39
   28 IF((NGPS-LGPNS).GE.MAXKC)29,31
C.... MAXKC CONSECUTIVE SATISFACTORY ESTIMATES.
   29 IF(LGPNS)30,35
C.... BACKWARD SMOOTHING NECESSARY.
   30 NGPS1=NGPS-KCBEGIN  $  NLRSR=NLRSR-KCBEGIN  $  GO TO 33
   31 IF(NGPS.GE.MAXS)32,39
   32 NGPS1=NGPS
   33 KONE=-1
   34 NGPS2=NGPS  $  KS=-1
      NRSEQ=.NOT.NRSEQ  $  NRRAN=.NOT.NRRAN  $  GO TO 39
C.... BACKWARD SMOOTHING NOT NECESSARY.
   35 NGPS1=1  $  NLRSR=NLRSR-NGPS+1  $  GO TO 34
C.... AN UNSATISFACTORY ESTIMATE.
   36 NNSAT=NNSAT+1  $  LGPNS=NGPS
   37 IF(KS.EQ.1)31,39
C.... EXTRACT DIAGONAL PORTION OF COVARIANCE MATRIX.
   39 DO 390 I=1,IST
  390 D(I)=P(I+IP*(I-1))
      CALL MATRIX(22,IST,1,0,XE,IST,XT,IST,XD,IST)
C.... PRINT SELECTED INFORMATION ON OUTPUT FILES.
      O(1)=TIME  $  GO TO M1,(40,41,55)
   40 CALL MOVLEV(XE(IS1),O(2),IS2)  $  CALL WRITER(1,O,IS3)
   41 GO TO M2,(42,43,44,46)
   42 CALL MOVLEV(X(IS1),O(2),IS2)  $  GO TO 45
   43 CALL MOVLEV(XD(IS1),O(2),IS2)  $  GO TO 45
   44 CALL MOVLEV(DX(IS1),O(2),IS2)
   45 CALL WRITER(2,O,IS3)
   46 GO TO M3,(47,48,51,52)
   47 CALL MOVLEV(D(IS1),O(2),IS2)  $  L=IS3  $  GO TO 50
   48 L=1  $  DO 49 I=IS1,IS4  $  DO 49 J=IS1,IS4  $  L=L+1
   49 O(L)=P(I+IP*(J-1))  $  GO TO 50
   50 CALL WRITER(3,O,L)
   51 CONTINUE
   52 GO TO M4,(53,54,55)
   53 IF(K)54,55
   54 CALL MOVLEV(NGPS,O(2),7)  $  GO TO M41,(540,541)
  540 K=0  $  IF(IOB.EQ.0) GO TO 542
      CALL MOVLEV(OBTYPES,O(9),2)  $  CALL MOVLEV(Y,O(11),IOB)
      K=2+IOB  $  GO TO 542
  541 CALL MOVLEV(IS,O(9),K)
  542 CALL WRITER(4,O,8+K)
   55 IF(NPRAN.AND..NOT.NWRAN)56,57
C.... SAVE DATA FOR EVENTUAL PLOTTED OUTPUT.
   56 CALL MOVLEV(XE,O(2),IST)  $  I=2+IST
      CALL MOVLEV(X,O(I),IST)  $  I=I+IST
      CALL MOVLEV(XD,O(I),IST)  $  I=I+IST
      CALL MOVLEV(D,O(I),IST)  $  I=I+IST-1
      CALL PACK1(I,O,MPT)
   57 RETURN  $  END
```

```
      SUBROUTINE FILEIO(T), RETURNS(NR1,NR2)
C.... THIS ROUTINE IS THE CONTROLLING ROUTINE FOR ALL FILE INPUT AND
C  OUTPUT IN THE PROGRAM.
      COMMON/STREF/IXR,XR(1)
      COMMON/STESTM/IXE,XE(1)
      COMMON/STCOV/IP,JP,P(1)
      COMMON/STCOVD/IPD,D(1)
      COMMON/RANFILE/MAXMS,MS(1)
      COMMON/PACKED/NLRS,NWORDS,WW(512)
      EQUIVALENCE (WW,IWW1),(WW(2),IWW2)
      COMMON/PARAMS/KSTRSDL,HSTEP,TIMETOL,KPUNCH,KREWIND,LRWDCNT(2),
     1 KPLOT,KEND,KONE,KS,KSMOOTH,MAXS,MAXKC,MPTY,NGPS,NLRSR,NLRSW,
     2 NFILF,NFILB,NSAT,NNSAT,NGPS1,NGPS2,NRSEQ,NRRAN,NWSEQ,NWRAN,NPRAN,
     3 IOB,JZH,IST,Z1(22),KQKLOOK
C.... THE FOLLOWING COMMON AND EQUIVALENCE STATEMENTS ARE DEPENDENT ON
C  THE DATA FILE USED.
      COMMON/BUFF/LENGTHR,LENGTHW,MAXLRW,LRW,W(127)
      EQUIVALENCE (W,TIME)
      COMMON RECORD(14)
      LOGICAL NRSEQ,NRRAN,NWSEQ,NWRAN,NPRAN
      DATA KOPEN/-1/
C.... SET FILE INPUT/OUTPUT OPTIONS.
      IF(KSMOOTH.NE.0.OR.KPLOT.NE.0) KOPEN=KOPEN+1
      IF(KOPEN)101,102,101
  102 CALL OPENMS(20,MS,MAXMS+1,0)  $  KOPEN=KOPEN+1
  101 MAXS=MIN0(MAXS,MAXMS)  $  MAXKC=MIN0(MAXKC,MAXS)
      IF(KREWIND.NE.0) REWIND 10
      ASSIGN 36 TO M1
      KREWRIT=0
      RETURN
      ENTRY FILEIO1
C.... RANDOM FILE READ.
  100 IF(NRRAN)1,15
    1 IF(KS)2,8,15
    2 IF(NGPS1.EQ.1)3,6
C.... BACKWARD SMOOTHING EITHER NOT NECESSARY OR NOW COMPLETE.
    3 KONE=1  $  KS=0  $  ASSIGN 37 TO M1
      NWSEQ=.NOT.NWSEQ  $  NWRAN=.NOT.NWRAN  $  NPRAN=.NOT.NPRAN
      IF(NGPS1.EQ.NGPS)37,10
C.... CONTINUE WITH BACKWARD SMOOTHING.
    6 IF(NGPS.EQ.NGPS1)9,7
    7 ASSIGN 11 TO M1  $  GO TO 10
    8 IF(NGPS.NE.NGPS2) GO TO 9
C.... RESUME SEQUENTIAL FILE READ.
      NRRAN=.NOT.NRRAN  $  NRSEQ=.NOT.NRSEQ
      KREWRIT=-1  $  GO TO 11
C.... READ IN A LOGICAL RECORD.
    9 NGPS1=NGPS1+KONE  $  NLRSR=NLRSR+KONE
   10 CALL READMS(20,LRW,MAXLRW,NGPS1)
      LRW=LRW-2*IST
      NGPS=NGPS1  $  GO TO M1,(11,36,37)
C.... MOVE APPROPRIATE VALUES FROM W ARRAY TO STATE AND COVARIANCE
C  ARRAYS
   11 I=LRW+1  $  CALL MOVLEV(W(I),XE,IST)
      I=I+IST  $  CALL MOVLEV(W(I),D,IST)
      CALL DIAGNAL(IST,XE,XR,D,IP,P)  $  T=TIME
      ASSIGN 36 TO M1  $  GO TO 100
```

```
C.... SEQUENTIAL FILE READ.
   15 IF(NRSEQ)16,22
C.... READ IN A LOGICAL RECORD.
   16 READ(10) LRW,(W(I),I=1,LRW)  $  NLRSR=NLRSR+1
C.... CHECK LAST BINARY READ.
      IF(IO EOF PE(10))24,17,15
   17 NLRSR=NLRSR-1
   22 KEND=1  $  RETURN NR2
C.... DATA LOGICAL RECORD.
   24 IF(TIME.LT.T-TIMEDL)49,35
C.... ACCEPTABLE DATA RECORD.
   35 NGPS=NGPS+1
   36 RETURN NR1
      ENTRY FILEIO2
C.... RANDOM FILE WRITE.
   37 IF(NWRAN)40,41
   40 I=LRW+1  $  CALL MOVLEV(XE,W(I),IST)
      I=I+IST  $  CALL MOVLEV(D,W(I),IST)
      LRW=LRW+2*IST
      CALL WRITMS(20,LRW,LRW+1,NGPS,KREWRIT)  $  GO TO 49
C.... SEQUENTIAL FILE WRITE.
   41 IF(NWSEQ)42,45
   42 I=LENGTHR+1  $  CALL MOVLEV(XE,W(I),IST)
      I=I+LENGTHW  $  CALL MOVLEV(D,W(I),IST)
      LRW=LENGTHR+2*LENGTHW
      WRITE(11) LRW,(W(I),I=1,LRW)  $  NLRSW=NLRSW+1
C.... OUTPUT DATA FOR EVENTUAL PLOTTING, COMPARISON, ETC.
   45 IF(NPRAN)46,49
   46 IF(MPTY.EQ.1)48,49
   48 M=NLRS+1
      CALL WRITMS(20,WW,NWORDS,M,KREWRIT)  $  NWORDS=0
      IF(M.EQ.MAXMS) NPRAN=.NOT.NPRAN
   49 IF(KEND.EQ.1)22,10
      ENTRY FILEIO3
C.... PERFORM END PROCESSING.
   50 FORMAT(*-MINIMUM AND MAXIMUM VALUES FOR DATA ON PLOT FILE.*T70*WOR
     1DS PER GROUP*I4,T100*TOTAL LOGICAL RECORDS*I4/)
   51 FORMAT(I4,2X,1P2E20.12)
   53 IF(KPLOT)54,56
   54 IF(NWORDS.EQ.0) GO TO 55  $  M=NLRS+1
      CALL WRITMS(20,WW,NWORDS,M,KREWRIT)
   55 CALL PACK2(N,W,MPTY)
      PRINT 50,IWW1,IWW2  $  DO 550 I=1,IWW1  $  M=I+2  $  N=M+IWW1
  550 PRINT 51,I,WW(M),WW(N)
      CALL WRITMS(20,WW,NWORDS,1,KREWRIT)
   56 IF(KQKLOOK)57,62
   57 DO 61 I=1,4
      ENDFILE I  $  BACKSPACE I  $  BACKSPACE I
      READ(I,58) RECORD
   58 FORMAT(13A10,A7)
      IF(IO EOF PE(I))59,61,61
   59 PRINT 60,I
   60 FORMAT(*-FILE*I2* LAST RECORD*/)
      PRINT 58,RECORD
   61 CONTINUE
   62 RETURN  $  END
```

N IOEOFPE 74/74 OPT=1 FTN 4.5+414

```
      FUNCTION IO EOF PE(J)
C.... THIS FUNCTION CHECKS FOR NORMAL, END-OF-FILE, AND PARITY-ERROR
C CONDITIONS ON UNIT J FOLLOWING A BINARY READ.
      DATA KOUNT/0/
      KOUNT=KOUNT+1
C.... NORMAL.
      K=-1
      IF(EOF(J))10,11
C.... END OF FILE.
   10 K=0  $  PRINT 20  $  GO TO 13
   11 IF(IOCHEC(J))12,14
C.... PARITY ERROR IN OR AFTER RECORD INDICATED.
   12 K=1  $  PRINT 21
   13 PRINT 22,J,KOUNT
   14 IO EOF PE=K
   20 FORMAT(24H0***** END OF FILE *****)
   21 FORMAT(25H0***** PARITY ERROR *****)
   22 FORMAT(* UNIT *I2,10X* RECORD *I4)
      RETURN  $  END
```

E OBSERVE 74/74 OPT=1 FTN 4.5+414

```
      SUBROUTINE OBSERVE
C.... THIS ROUTINE SELECTS THE TYPES OF DATA REQUESTED FROM THE DATA
C  FILE AND PLACES THEM IN AN INCREASING SEQUENCE IN STORAGE FOR USE BY
C  PROGRAM ESTMATE.
      COMMON/SENSORS/SEMA(18),COILS(3,2)
      COMMON/OBSH/IH,JH,H(1)
      COMMON/PARAMS/IQ(33),KDATA,KTYPES,MAXDATA,IA,JA,KA,IM,JM,KM,MD
C.... THE FOLLOWING COMMON STATEMENT IS DEPENDENT ON THE DATA FILE USED.
      COMMON/BUFF/LBUFF(4),W(127)
      DIMENSION DATA(1),IDATA(1)
      EQUIVALENCE (W(19),ND),(W(20),DATA,IDATA)
      COMMON/TORQUES/COIL(3),TORQMAG(3),DENCOEF
      COMMON D(36),T(20),LOC(20),MM(4),LM1(4)
      DIMENSION LM2(4),LM3(4)
      DIMENSION COILSP(3,2)
      DATA COIL,COILSP/9*0./
      DATA KMASK/77777777777777777774B/
      LM2(3)=LM2(2)=LM2(1)=JM  $  LM3(3)=LM3(2)=LM3(1)=KM
      LM2(4)=JA  $  LM3(4)=KA
      RETURN
      ENTRY OBSERV1
C.... SELECT DATA FROM DATA FILE.
      KDATA=K=MM(4)=MM(3)=MM(2)=MM(1)=0
      KCOUNT=0
      LM1(3)=LM1(2)=LM1(1)=IM  $  LM1(4)=IA
   20 IF(K.EQ.ND) GO TO 31
      J=MD*K+1  $  K=K+1
      I=KTYPE=IDATA(J+2)  $  IF(I.GT.2) I=I.AND.KMASK
      IF(I.AND.KTYPES)21,20
   21 IF(KTYPE.AND.8)22,230
```

```
C.... STORE COIL DATA.
   22 L=KTYPE-7  $  DO 23 I=1,3
   23 COILSP(I,L)=DATA(J+1)*COILS(I,L)
      CALL MATRIX(21,3,1,0,COILSP,3,COILSP(4),3,COIL,3)  $  GO TO 20
  230 IF(KTYPE.AND.16) 231,24
C.... SAVE THE ONE-HALF-RHO-V-SQUARED.
  231 DENCOEF=DATA(J+3)  $  GO TO 20
C.... SAVE OTHER TYPES OF DATA FOR PROCESSING.
   24 IF(KTYPE.LE.2) GO TO 26  $  L=MOD(KTYPE,9)-3
      MM(L)=MM(L)+1  $  IF(MM(L).EQ.LM1(L))25,20
   25 LM1(L)=LM1(L)+LM3(L)  $  IF(LM1(L).GT.LM2(L)) LM1(L)=0
   26 KDATA=KDATA+1  $  T(KDATA)=DATA(J)  $  LOC(KDATA)=K
      KCOUNT=KCOUNT+1  $  IF(KTYPE.EQ.1) KCOUNT=KCOUNT+2  $  GO TO 20
C.... LIMIT DATA IF NECESSARY.
   31 L=KCOUNT-MAXDATA  $  IF(L.GT.0) KDATA=KDATA-L
C.... RETURN WHENEVER NO DATA IS SELECTED OR NO ORDERING OF DATA IS
C  NEEDED.
      IF(KDATA-1)40,38,310
C.... ORDER THE DATA IN AN INCREASING SEQUENCE BY TIME.
  310 DO 37 I=2,KDATA  $  J=I-1
      TEST=T(I)  $  IF(TEST.GE.T(J)) GO TO 37  $  LO=LOC(I)  $  L=I
   32 J=J-1  $  IF(J)33,34
   33 IF(TEST.LT.T(J))32,34
   34 J=J+1
   35 LM1=L-1  $  T(L)=T(LM1)  $  LOC(L)=LOC(LM1)
      L=L-1  $  IF(L.EQ.J)36,35
   36 T(J)=TEST  $  LOC(J)=LO
   37 CONTINUE
C.... STORE THE SELECTED DATA.
   38 DO 39 I=1,KDATA  $  L=MD*(LOC(I)-1)+1  $  J=MD*(I-1)+1
   39 CALL MOVLEV(DATA(L),D(J),MD)
   40 RETURN  $  END
```

```
E PACK          74/74    OPT=1                                    FTN 4.5+414

      SUBROUTINE PACK(N,V,M)
C.... THIS ROUTINE STORES DATA BEFORE IT IS OUTPUT AS A SINGLE LOGICAL
C  RECORD.  THE RANGE OF THE DATA IS ALSO COMPUTED.
      DIMENSION V(1),VMIN(50),VMAX(50)
      COMMON/PACKED/NLRS,NWORDS,VV(512)
      EQUIVALENCE (VV(1),IWD),(VV(2),J)
C.... MAXIMUM NUMBER OF WORDS IN A LOGICAL RECORD IS 512.
C.... MAXIMUM NUMBER OF WORDS IN A GROUP IS 50.
      DATA MAXD/512/,MAXWD/50/
C.... INITIALIZE PARAMETERS.
      NLRS=0
      IWD=N=MIN0(N,MAXWD)   $  NGP=(MAXD-2)/IWD
      DO 1 K=1,IWD  $  VMIN(K)=1.E99
    1 VMAX(K)=-1.E99
      NWORDS=J=0
      RETURN
      ENTRY PACK1
C.... STORE THE DATA.
      IF(M.EQ.1) J=0
      IF(J.NE.0) GO TO 3  $  M=0  $  NLRS=NLRS+1
    3 J=J+1  $  N=MIN0(N,IWD)  $  L=IWD*(J-1)+2
      DO 2 K=1,N
      VT=V(K)  $  IF(VT.LT.VMIN(K)) VMIN(K)=VT
      IF(VT.GT.VMAX(K)) VMAX(K)=VT
    2 VV(L+K)=VT
      IF(J.EQ.NGP) M=1
      NWORDS=L+IWD
      RETURN
      ENTRY PACK2
C.... READ OUT THE STORED RANGE VALUES.
      NWORDS=2*IWD+2  $  J=NLRS
      CALL MOVLEV(VMIN,VV(3),IWD)  $  CALL MOVLEV(VMAX,VV(3+IWD),IWD)
      RETURN  $  END
```

```
E DIAGNAL       74/74    OPT=1                                    FTN 4.5+414

      SUBROUTINE DIAGNAL(N,X,XX,D,IP,P)
C.... UTILITY SUBROUTINE TO MOVE THE N-DIMENSIONAL STATE VECTOR X INTO
C  XX AND TO MAKE THE N-DIMENSIONAL VECTOR D THE DIAGONAL MATRIX P.
C.... NOTE.  IP IS COLUMN SIZE OF MATRIX P.
      DIMENSION X(1),XX(1),D(1),P(1)
      CALL MOVLEV(X,XX,N)
      DO 2 I=1,N  $  L=IP*(I-1)
      DO 1 K=1,N  $  J=K+L
    1 P(J)=0.
      J=I+L
    2 P(J)=D(I)
      RETURN  $  END
```

References

1. Sodano, E.M. (1958) A Rigorous Non-Iterative Procedure for Rapid Inverse Solution of Very Long Geodesics, Bulletin Geodesique, Issues 47/49, 13-25.

2. Sodano, E.M. (1965) General Non-Iterative Solution of the Inverse and Direct Geodetic Problems, Bulletin Geodesique, No. 75, 69-89.

3. Hufford, G.A. (1952) An integral equation approach to the problem of wave propagation over an irregular surface, Quart. Appl. Math 9:391.

4. Johler, J.R. (1971) LORAN Radio Navigation Over Irregular, Inhomogeneous Ground With Effective Ground Impedance Maps, Telecommunications Research and Engineering Report 22 (Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402).

5. Johler, J.R., and Horowitz, S. (1975) Propagation of a LORAN Pulse Over Irregular, Inhomogeneous Ground, AGARD-CP-144 (The Hague, Netherlands Conference, March 1974).

6. Spies, K.P. (1975) The Analytical Basis of Hufford's Computer Technique for Determining Topographic Profiles, Institute for Telecommunication Services Memo dated Nov. 4, 1975.

7. Sheed, F. (1962) Theory and Problems of Numerical Analysis, Shaum's Outline Series, McGraw Hill Co.

8. Johler, J.R., and Horowitz, S. (1973) Propagation of LORAN-C Ground and Ionospheric Wave Pulses, Office of Telecommunications Report 73-20 (Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402).

9. Johler, J.R., Keller, W.J., and Walters, L.C. (1956) Phase of the Low Radiofrequency Ground Wave, NBS Circular 573 (Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402).

10. Abramovitz, M. and Stegun, I. (1964) Handbook of Mathematical Functions — NBS — Applied Math Series 55, U.S. Government Printing Office.

11. Scott, R. (1966) Phase of the Height-Gain Function of the Low Frequency Ground Wave, Report 2900-156-T of Project Michigan (Willow Run Laboratories, Ann Arbor, Michigan 48106).

12. Johler, R.J. (1977) Prediction of Ground Wave Propagation Time Anomalies in the LORAN C Signal Transmission Over Land, AGARD Conference Proceedings, No. 209.

13. Horowitz, S., and Johler, R.J. (1977) LORAN C/D Coordinate Prediction Dependence on Ground Electrical Properties, AGARD Conference Proceedings, No. 209.

14. Davidson, D., Macklin, D., and Vozoff, K.V. (1974) Resistivity surveying as an aid in sanguine site selection, IEEE Transactions on Communications, Com 22:(No. 4).

15. Langer, R.E. (1936) On the determination of earth conductivity from observed surface potentials, Amer. Math. Soc. Bull. 42:747.

16. Kunetz, G. (1966) Principles of Direct Circuit Resistivity Prospecting, Berlin: Borntraeger.

17. Wait, J.R. (1970) Electromagnetic Waves in Stratified Media, 2nd Ed., New York: Macmillan.

18. Kunetz, G. (1972) Processing and interpretation of magnetotelluric soundings, Geophysics 37:1005.

19. Flathe, H. (1976) The Role of a Geologic Concept in Geophysical Research Work, Geoexploration 14 195-206 Elsview Pub. Co., Amsterdam, Netherlands.

20. Stoyer, C.H., and Wart, J.R. (1977) Resistivity Probing of an Exponential Earth With a Homogeneous Overburden, Geoexploration 15 11-18.

21. Hanle, E. (1966) The Complex Impedance of the Earth's Surface at Radio Frequencies and Its Measurement, NTZ-CJ, No. 3, West Germany.

22. Elkins, T.E. (1976) Empirical Correction of Soil Conductivity Model, RADC/ET Private Communication.

23. Program ESTIMAE - Problem No. 4723 (1975) Analyses and Simulation Branch, AFCRL dated 1 November 1975.

Conductivity Bibliography

1. Barfield, R.H. (1928) The attenuation of wireless waves over land, Jour. IEE. 66:204.

2. Barfield, R.H. (1928) Some measurements of the electrical constants of the ground at short wavelengths by the wave-tilt method, Jour. IEE. 66.

3. Card, R.H. (1935) Earth resistivity and geological structure, Trans. Amer. Inst. Elec. Engrs. 54:1153-1161.

4. Card, R.H. (1937) Correlation of earth resistivity with geological structure and age, Am. Inst. Met. Engr. Tech. Pub. 829.

5. Challinor, J. (1967) A Dictionary of Geology, Oxford Press.

6. Cherry, R.O. (1930) Field intensity measurements around some Australian broadcast stations, Proc. of the Physical Soc. of London 42:192.

7. Collard, J. (1932) Measurements of mutual impedance of circuits with earth return, Jour. IEE. 71:674.

8. Electrical Characteristics of the Surface of the Earth (1974) C.C.I.R. Report 229-2, XIII of Plenary Assembly, Geneva.

9. Frohlich, R.U. (1964) Geoelectrical measurements on a fault in the tertiary basin of Maine (Germany) using the four point method, Geoexploration 2:175.

10. Gish, O.H., and Rooney, W.J. (1925) Measurement of resistivity of large masses of undisturbed earth, Terr. Mag. and Atmospheric Elec. 30:161-188.

11. Heiland, C.A. (1946) Geophysical Exploration, Prentice-Hall.

12. Higgs, P.J. (1930) An investigation of earthing resistances, Jour. IEE. 68:736.

13. Keller, G.V. (1959) Analysis of some electrical transient measurements on igneous, sedimentary and metamorphic rock, in Overvoltage Research and Geophysical Applications, Pergamon Press, p. 92.

14. Keller, G.V. (1966) Electrical Methods in Geophysical Prospecting, Pergamon Press.

15. Lancaster-Jones, E. (1930) The earth resistivity method of electrical prospecting, The Mining Magazine.

16. Lowy, H. (1911) Dielectric constant and conductivity of rocks, Annalen der Physik 36:125.

17. Mielecke and Walter (1956) Geoelectrical measurements as an aid to geologic mapping, Zeitschur, Angew. Geologic 2:154.

18. Moore, R.W. (1945) An empirical method of interpretation of earth resistivity measurements, Trans. A.I.M.E. 164:197.

19. Morgan, R.R. and Maxwell,(1965) Omega Navigational System Conductivity Map, Rep. No. 54-F-1, DECO Electronics, Inc.

20. Nosske, Gerhard, and Franke (1958) Some experiences in the geoelectric mapping of mineral veins, fissures and faults in the Mittelgebirge, Zeitsch Geophysik 24:340.

21. Palmer, L.S. (1959) Examples of geoelectric surveys, Proc. IEE 106A:231.

22. Parkhomenko, E.I. (1967) Electrical Properties of Rocks, Plenum Press.

23. Pomper, Johannes, Frahlich, and Lother (1960) Extension of electrical surveying to the investigation of glaciodynamically deformed clays, Zeitschr. Argew. Geologic 6:(No. 8)387.

24. Ratcliffe, J.A. and White (1930) The electrical properties of the soil at radio frequencies, Philosophical Magazine 10:667.

25. Rooney, W.J. and Gish (1927) Results of earth resistivity surveys near Watheroo, Western Australia and at Ebro, Spain, Terr. Mag. and Atmospheric Elec. 32:49.

26. Rooney, W.J. (1927) Earth resistivity measurements in the Copper Country Michigan, Terr. Mag. and Atmospheric Elec. 32:97.

27. Roman, I. (1931) Electrical Resistivity of Underlaying Beds, Bureau of Mines Tech Paper 502.

28. Simpson, R. (1966) Rocks and Minerals, Pergamon Press.

29. Smith-Rose, R.L. and McPetrie (1931) The attenuation of ultra short radio waves due to the resistance of the earth, Proc. of the Physical Soc. of London 43:592.

30. Smith-Rose, R.L. and McPetrie (1932) The propagation along the earth of radio waves on a wavelength of 1.6 meters, Proc. of the Physical Soc. of London 44:500.

31. Smith-Rose, R.L. (1933) The electrical properties of soil for alternating currents at radio frequencies, Proc. of the Physical Soc. of London 140:359.

32. Smith-Rose, R.L. (1934) Electrical measurements on soil with alternating currents, Jour. IEE. 75.

33. Tagg, G.F. (1931) Practical investigations of the earth resistivity method of geophysical surveying, Proc. of the Physical Soc. of London 43:305.

34. Tagg, G.F. (1964) Earth Resistances, Pitman Pub. Corp.

35. Watt, A.D., Mathews and Maxwell (1963) Some electrical properties of the earth's crust, Proc. IEEE 51:(No. 6).

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.

Printed by
United States Air Force
Hanscom AFB, Mass. 01731

78